SYSTEMS ENGINEERING
Research Center

RT 128: New Project Incubator

Technical Report No. SERC-2015-TR-106

September 4, 2015

Principal Investigator: Dr. Jon Wade, Stevens Institute of Technology

Mark S. Avnet, Texas A&M

LiGuo Huang, Southern Methodist University

Azad Madni, University of Southern California

Kevin Sullivan, University of Maryland

Gary Witus, Wayne State

Copyright © 2015 Stevens Institute of Technology

The Systems Engineering Research Center (SERC) is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

This material is based upon work supported, in whole or in part, by the U.S. Department of Defense through the Office of the Assistant Secretary of Defense for Research and Engineering (ASD(R&E)) under Contract HQ0034-13-D-0004.

Any views, opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense nor ASD(R&E).

No Warranty.
This Stevens Institute of Technology and Systems Engineering Research Center Material is furnished on an “as-is” basis. Stevens Institute of Technology makes no warranties of any kind, either expressed or implied, as to any matter including, but not limited to, warranty of fitness for purpose or merchantability, exclusivity, or results obtained from use of the material. Stevens Institute of Technology does not make any warranty of any kind with respect to freedom from patent, trademark, or copyright infringement.

This material has been approved for public release and unlimited distribution.

TABLE OF CONTENTS

1 INTRODUCTION

As described in this Technical Plan, the SERC performs research on 20-25 active tasks on well-defined topics that are aligned with the SERC's research strategy. While it is believed that that the aforementioned research programs have a great potential to have a transformative impact on the DoD and IC, there is a need to support new ideas in their infancy that may become the critical research programs for emerging challenges. This incubation capability will be supported by an annual open call to the SERC research collaborating universities to propose early stage research that can be nurtured through relatively small levels of seed funding.

The initial open call took place in September 2014 with the objective of identifying and developing several short white papers outlining research programs with a significant potential to improve the practice of engineering systems. A total of 29 responses were received as shown in Table 1.1 below:

Table 1.1: Responses to 2015 SERC Incubation Grant Solicitation

University	PI	Team	Title
Auburn University	Levent Yilmaz	Alice Smith	A Domain Architecture and Generative System for Feature Driven Conceptual Modeling of Family of Systems
Carnegie Mellon University	Jonathan Aldrich	Joshua Sunshine	Capability-Based Modules for Architectural Control
Georgia Institute of Technology	Dimitri Mavris		"Development of an Agile Systems Engineering Environment to Support Flexible COTS-Based Systems Engineering
Georgia Institute of Technology	Dimitri Mavris		Open Systems Architecture Initiative
Missouri S&T University	Renzhong Wang	Cihan Dagli	Computational System Architecture Development using Holistic Modeling|
Missouri S&T University	Cihan Dagli	George Muller	Modeling Coevolution in Systems-of-Systems Architectures
Missouri S&T University	Charles O. Adler	Cihan Dagli	Networked Systems of Systems Architectures: Robustness Evaluation and Improvements Using Dynamic Failure Valuation and Genetic Algorithms
Missouri S&T University	Dincer Konus	Cihan Dagli	Stability of System of Systems Architecture
Naval Postgraduate School	Kristin Giammarco	Mikhail Auguston	A common modeling framework to PM and System Architects for describing and analyzing possible behaviors of a system under design
Naval Postgraduate School	Warren K. Vaneman	Kostas Trantis (Virginia Tech)	Designing Resiliency into Department of Defense critical Systems
North Carolina A&T State University	William Edmonson	william.edmonson@nianet.org	System of Systems Design and Verification Using Formal Methods and Multi-Disciplinary Simulation
Purdue University	Monica E. Cardella	John Mendoza-Garcia	Developing an Empirical Framework for Characterizing Different Levels of Systems Thinking

Southern Methodist University	LiGuo Huang		Collaborative Knowledge Learning and Transfer over Systems
Southern Methodist University	LiGuo Huang	JoAnn Lane, USC	Detecting and Evaluating Technical Debt of Software Systems
Stevens Institute of Technology	Babak Heydari		Dynamic Structures for Distributed Systems with Heterogeneous, Autonomous Constituents
Stevens Institute of Technology	Babak Heydari	Paul Collopy, UAH	Theory of Modularity for Complex Systems with Rapidly evolving Capabilities
Stevens Institute of Technology	Rob Cloutier	Brian Sauser, University of North Texas	Transitioning Systems Thinking to Model-Based Systems Engineering
Texas A&M University	Mark Avnet	Tonya L. Smith-Jackson, NCAT	"A Multilevel Framework of System Safety: Technical Failures, Human Factors, Organizational Culture, and Societal Influence"
University of Alabama Huntsville	Paul Collopy		Systems Engineering and Digital Manufacturing
University of Alabama Huntsville	Paul Collopy		Verification of Complex Cyberphysical Systems
University of Massachusetts	Lance Fiondella		Incorporating Reliability Engineering into Tradespace Analysis and Exploration
University of Southern California	Azad M. Madni		Formal Methods in Resilient Systems Design using a Flexible Contract Approach
University of Texas at Austin	Armand J. Chaput		Systems Engineering Design - How to teach SE in capstone design without displacing other course content
University of Virginia	Barry Horowitz	Dan DeLaurentis, Purdue	Developing Requirements and Architectures for "Successfully" Autonomous Systems
University of Virginia	Kevin Sullivan	John Baras, UMD	Foundations of Systems Engineering
Wayne State	Gary Witus		Engaging End-Users In SE Research Planning and Transition
Wayne State	Gary Witus	Mark Blackburn (Stevens)	Policies and Practices for Model-Centric Government-Industry Collaborative Environments for Systems Engineering and Development
Wayne State	Gary Witus	Mark Blackburn (Stevens), Barry Boehm (USC)	Pre-Acquisition SE for Software-Intensive Cyber-Physical Systems

.

Each of the received proposals were reviewed by the Research Council except where they noted that they had a potential conflict. Each member provided a final score based on an equal weighing in each of the following four criteria as well as a set of short comments:

- Intellectual Merit
- Clarity of Vision
- Past Performance
- Potential Strategic Impact

Each of the Research Council member's ratings were normalized based on their average numerical score and an average score was calculated for each proposal. To aid in the visualization of the results, the rankings for each Research Council member were color coded to be Green – top 1/3, Yellow – middle 1/3 and Red – bottom 1/3. The proposals were ranked based on these results.

The sponsor independently ranked the top ten projects based on these criteria. Then the PI met with the sponsor and the proposals were discussed. Preference was given to proposals that contend with issues not currently being addressed by SERC research, or use novel approaches, but which support the current SERC UARC research focus areas and have a strong potential for additional funding outside of SERC core funds. In this case, the selected six proposals were in the top ten proposals as determined by the Research Council.

Of these, the following six proposals were selected:

1. Mark S. Avnet, Texas A&M, *A Multilevel Framework of System Safety: Technical Failures, Human Factors, Organizational Culture, and Societal Influence*

2. LiGuo Huang, SMU, *Detecting and Evaluating Technical Debt of Software Systems*

3. Azad Madni, USC, *Formal Methods in Resilient Systems Design using a Flexible Contract Approach*

4. Kevin Sullivan, UVa, *Foundations of Systems Engineering*

5. Gary Witus, Wayne State, *Policies and Practices for Model-Centric Government-Industry Collaborative Environments for Systems Engineering and Development*

6. Rob Cloutier, Stevens Institute of Technology, *Transitioning Systems Thinking to Model-Based Systems Engineering*

Unfortunately, contractual issues did not allow for the funding of Cloutier's proposal.

This report contains the white papers that was supported by each of these funded proposals. Presentations of this work is contained in the Appendix of this report.

2 A MULTILEVEL FRAMEWORK OF SYSTEM SAFETY: TECHNICAL FAILURES, HUMAN FACTORS, ORGANIZATIONAL CULTURE, AND SOCIETAL INFLUENCE - MARK S. AVNET, TEXAS A&M, NCA&T

2.1. OBJECTIVES OF THE RESEARCH

In recent years, it has become widely accepted that accidents involving complex technologies can rarely be attributed to technical causes alone. Still, the patterns of interactions among the many underlying contributing factors of such accidents are not well understood. As Reason argues, these "latent failures" are defined by their presence "within the system well before the onset of a recognizable accident sequence" (1990). Until recently, most research on system safety focused on simplistic (even if mathematically complicated) reliability models that assess safety by multiplying probabilities of component failures. In many settings, the goal of accident investigations still is to complete a root cause analysis to identify a key decision point that can explain a chain of events leading to an accident. As observed by the Columbia Accident Investigation Board (CAIB), these investigations often stop at placing blame on a specific technical flaw, person, or procedure (2003). Research examining the role of human error, however, only scratches the surface of uncovering the true causes of accidents involving complex systems. Most difficult safety-related problems in these contexts have more to do with organizational systems and structures than with specific technical issues or human mistakes, and the proximate technical cause of an accident is often only a symptom of a broader and more systemic problem (Leveson, 2011). Even when research and accident investigations do probe more deeply to examine underlying organizational causes of an accident, the complex interactions among technical, human, organizational, and societal elements of the system often are not considered. Thus, investigations usually do not lead to system-level changes that prevent future accidents. Accident prevention requires a truly holistic perspective that extends beyond the organizational level to also include the broader societal, policy, and regulatory environment as part of the system. The resulting analysis cannot merely lead to broadly based models of the control structure but needs to incorporate the nuances at all levels of analysis while still providing a means to understand the interconnections across those levels.

The purpose of this white paper is to present a developing research program intended to advance the theory and practice of system safety by explicitly examining the causes of accidents from a multilevel perspective, including proximate technical causes, human error, organizational culture, and societal influences. A growing body of literature on systems approaches to safety has laid the foundation for an interdisciplinary effort examining the nature of the relationships across these levels. At each level of the hierarchy, a distinct and rich set of methods, tools, and disciplinary knowledge exists and has been applied to a wide array of system contexts including aerospace, aviation, transportation, chemicals, oil and gas, nuclear power, disaster response, healthcare, and others. Each of these disciplinary perspectives provides a unique lens to examine safety in complex engineered systems. For the first time, the field of system safety has advanced to a point at which both the disciplinary methods and the holistic systems perspective can be integrated and used to build a robust and comprehensive understanding of the depth, nuance, and complexity of system safety.

The overarching goal of the proposed research program is to advance the **theory and practice** of system safety by **integrating holistic systems approaches with the richness of disciplinary methods** to explicitly examine safety from **a multilevel perspective**, including proximate technical causes, human error, organizational culture, and societal influences. This goal will be achieved by addressing three more specific research objectives:

1. To develop a robust framework for ensuring system safety that offers **specific and practical guidelines** for system development, management, accident investigation, and policymaking from a **holistic, multilevel, and interdisciplinary** perspective
2. To create a universal **repository of safety-related knowledge** across disparate contexts, industries, and disciplines to serve as the **engine for implementing and operationalizing** the guidelines of the framework
3. To provide a **platform** for **fundamental research** focused on the **theoretical foundations** of each level of the framework

The research methodology, which is based on an examination of the literature of several fields combined with laboratory- and field-based studies, represents a first attempt to integrate perspectives, methods, tools, and domains within one framework. The repository, also called the System Safety Database, will provide a desperately-needed and long-overdue consolidated reference to the existing body of knowledge needed to implement the framework effectively. It will facilitate analysis across disciplines and contexts, allowing researchers and practitioners to use integrated mixed-methods approaches to conduct investigations, analyses, research, and development activities across multiple levels of a system. This work will lead to a fundamental understanding of the overall ecosystem of safety and will provide stakeholders at all levels, from individual operators to policymakers, with the tools and perspectives needed to improve system safety. Furthermore, while the scope of the present research is focused on safety, the methods and outcomes will be applicable to various dimensions of performance and assurance in our society's complex engineered systems.

2.2. Current Practice in System Safety and Its Limitations

The study of safety in complex engineered systems is not a new idea. An extensive body of literature on concepts related to system safety exists in each of several fields, from engineering to political economy, and focuses on a variety of contexts, ranging in scale from individual patients in healthcare settings to more than 500,000 victims in the 1984 Bhopal disaster. Over the years, a growing number of researchers have advocated systems-oriented approaches (e.g., Reason, 1997) and more recently have developed specific systems engineering-based techniques (e.g., Leveson, 2011). Prior disciplinary work has provided the analytical backbone for understanding safety, and systems approaches have offered a holistic viewpoint needed to examine interconnections among disciplines. Based on this work, the stage is now set for an integrative framework that explicitly links the findings from many disciplines and contexts to the broader system. The purpose of this section is to describe the current state of the art that forms the foundation for the proposed interdisciplinary multilevel framework of system safety.

2.2.1. Existing Frameworks, Theories, and Methods for Analysis of System Safety

Safety is a system property that has been an important focus of many distinct disciplines for several decades. Traditional quantitative methods for probabilistic risk assessment and a variety of design-based methods are used in engineering. Psychologists and human factors engineers apply cognitive psychology to understand the role of humans while social and industrial/organizational psychologists focus on the role of culture or climate in organizations. Many industries have adopted several of the above methods for application in their own contexts, and systems engineers have taken a more holistic view to study interaction effects that influence safety. In parallel, political scientists and economists have taken an even broader perspective by examining system safety in terms of the role of policy, regulation, and public perception of risk in various countries. Table 2.1 provides an overview of just a

Table 2.1. Selection of Frameworks, Theories, and Methods for Analysis of System Safety

Category	Method/Approach	Description	Reference(s)
Quantitative Techniques	*Quantitaive Risk Assesment (QRA)*	Traditional quantitative assessment to prioritize prevention or mitigation activities	Kennedy and Kirwan (1998); Apostolakis (2004)
	Root Cause Ananlysis (RCA)	Method to identify basic and causal factors of performance variation that rests on the premise of a single cause to prevent or control	Wu et al. (2008)
	Hazard and Operability (HAZOP) Study	Method to identify hazards in processes/facilities and how a process may deviate from its design intent	Kletz (1997); Dunjó et al. (2010)
Engineering Design Approaches	*Preliminary Hazard Analysis (PrHA)*	Largely qualitative method to identify hazards early in design using scenarios, brainstorming, legacy systems, and accident reports	Rausand (2004)
	Failure Mode and Effects Analysis (FMEA)	Bottom-up analysis of component failure modes and their effects on the system	Dyer et al. (1972)
	Fault Tree Analysis (FTA)	Top-down use of logic diagrams to portray and analyze potentially hazardous events	Lee et al. (1985); Hixenbaugh (1968)
	Inherently Safer Design (ISD)	Upfront consideration of safety in engineered systems to ensure that accidents cannot happen by design	Kletz (2003)
Human Factors Methods	*Skill-Rule-Knowledge (SRK) Model*	Model of human error using skill- (unconscious actions), rule- (conscious execution), and knowledge-based (problem-solving) cognition	Rasmussen (1983); Vicente (1999)
	User Centered Design (UCD)	Design for human use through usability testing/engineering, heuristic evaluation, discount evaluation, and participatory design	Norman (1986); Abras et al. (2004)
Organizational Frameworks	*Safety Climate*	Measurement and analysis of shared perceptions of an organization's policies, procedures, and practices in high-risk operations	Zohar (2010); Payne et al. (2009)
	Swiss Cheese Model	Framework of layers of barriers with "holes" representing latent and active failures such that an accident occurs when the "holes" line up	Reason (1997)
	Human Factors Analysis and Classification System (HFACS)	Tool for classification of human errors based on the Swiss Cheese model	Shappell and Wiegmann (2001)
	High Reliability Organization (HRO) Theory	Study of the characteristics of organizations that seldom experience system failures as compared to those of other organizations	Weick et al. (2008); Roberts (1990)
	Normal Accident Theory (NAT)	Perspective that the high complexity and tight coupling make some (but not all) occasional catastrophic accidents inevitable and "normal"	Perrow (1994); Perrow (2011)
Industry-Specific Approaches	*Crew Resource Management (CRM)*	Approach used in aviation to ensure that inadequate training, human imperfections, and poor communication do not lead to accidents	Cooper (1980)
	Process Hazard Analysis (PHA)	Core of Process Safety Management (PSM) involving "careful review ... to prevent releases of hazardous chemicals" (OSHA, 2000)	OSHA (2000); Baybutt (2003)
	Systems Engineering Initiative for Patient Safety (SEIPS)	Framework, extending the Donabedian structure-process-outcome model, to improve patient safety using a systems approach	Carayon et al. (2006); Donabedian (2005)
Systems Approaches	*System-Theoretic Accident Model and Process (STAMP)*	Technique for analyzing system safety using the principles of control theory	Leveson (2011)
Political Economy Theories	*Precautionary Principle*	Decision framework that places protection of health, safety, and environment above the need for conclusive scientific evidence of risk	Morgan (1993); Ashford (2007)
	Regulatory Capture	Condition in which private interests exert undue influence on regulatory bodies, which could lead to poor safety regulations	Stigler (1971); Levine et al. (1990)

sample of some of the many frameworks, theories, and methods used across disciplines in the study of various dimensions of system safety.

2.2.2. LIMITATIONS OF CURRENT METHODS

Current approaches to system safety have three general but highly connected limitations: (1) emphasis on *post hoc* accident investigation, (2) poor application of lessons learned from across industries and sectors, and (3) lack of a holistic multilevel approach. Each of these classes of limitations is described in this subsection.

Hindsight-Driven Analysis. One of the most significant challenges in ensuring safety in complex systems is the inherent lack of transparency into what could happen in the future. Indeed, major accidents happen in complex systems precisely because they are unpredictable. By definition, major system failures occur when they are not predicted and thus appropriate remedial actions are not taken. Although organizations generally do tend to learn lessons from specific failures and ensure that the same failures do not happen again, this backward-looking approach does not sufficiently capture the underlying systemic issues that make accidents happen. For example, Johns Hopkins Hospital Children's Center effectively eliminated central line-associated bloodstream infections (CLABSIs) following the 2001 death of 18-month-old Josie King (Pronovost and Vohr, 2010), yet the simple practices implemented at Johns Hopkins have not effectively diffused throughout the healthcare system (Liang and Marschall, 2011; Furuya, Dick, Perencevich, Pogorzelska, Goldman, and Stone, 2011). Similarly, the Rogers Commission's recommendations following the Space Shuttle *Challenger* explosion (1986) were implemented in NASA's human spaceflight program, ensuring that another temperature-related O-ring failure would not occur and that an independent safety oversight function was put in place. The underlying "can-do" NASA culture linked to the Cold War-related fervor of the Apollo program, however, was not fully addressed. As a result, NASA suffered yet another loss with the 2003 breakup of the Space Shuttle *Columbia*. While this accident resulted from a completely different proximate technical cause (foam shedding from the external tank), the underlying systemic causes of the *Columbia* disaster were found to be linked to essentially the same organizational factors that had existed 17 years earlier (CAIB, 2003). If decision makers in organizations had more convenient and ready access to lessons learned from a variety of contexts, their ability to proactively address a wide array of potential hazards and avoid such repeat events would be greatly improved.

Insufficient Use of Lessons Learned. It is perhaps an undisputed fact among the safety community that codification of data and lessons learned about system safety is woefully inadequate. In some cases, parallels are too quickly drawn across drastically different contexts. For example, one criticism of the theory of high reliability organizations (HROs) is that virtually all identified HROs operate in relatively contained contexts, such as Naval vessels and aircraft cockpits, while many accidents occur in less much larger organizations (Casler, 2014). Conversely, many important lessons that can be drawn across contexts are often ignored or simply lost. For example, preventable medical errors represent a particularly insidious problem because each incident generally leads to the injury or death of just one patient but, in sum, those incidents represent one of the top ten leading causes of death in the United States today (Kohn, Corrigan, and Donaldson, 2000). Because of the invisibility of this problem, many of the best practices from other industries have not been widely adopted in this context. Recently, attempts have been made to implement a modified form of aviation's crew resource management (CRM), but these efforts are slowed by a hierarchical professional culture in healthcare. Furthermore, when issues are identified in patient care settings, they are often buried and thus go unreported, making it difficult or impossible to learn from past mistakes within the healthcare system and much less from those made in other industries. A comprehensive database that captures not only the history of case studies across industries but also the wide array of analysis methods and tools brought to bear would improve the ability to apply lessons learned when appropriate while still avoiding the pitfalls associated with improperly translating interventions across industries.

Siloed Disciplinary Approaches. The final set of limitations of current approaches is less related to problems with particular methods but rather to the reality that they are generally used within disciplinary silos that mask the inter-relationships across different levels of accident causation (technical, human, organizational, and societal). Perspectives on safety and risk abound in nearly every area of study. As discussed in the previous subsection, these methods and approaches draw on insights from

fields as diverse as probabilistic modeling, engineering design, human factors, organizational behavior, systems theory, and political economy in addition to specific insights from particular industries and sectors. Still, existing methods do not integrate the advantages offered by all of these different approaches into one holistic framework that can be used to improve safety in real-world complex systems. While Reason's Swiss cheese model (1997) offers a valuable first step, the model's representation of protective layers and "holes" in each layer suggests a linearity in the relationships among factors and thus tends to oversimplify interaction effects. While Reason notes that the "holes" move over time, his model does not explicitly incorporate the mechanisms of that movement. Similarly, Reason discusses complex interactions and feedback in systems, but the Swiss cheese model does not explicitly address the effect that the position of one "hole" in the Swiss cheese can have on that of another. Furthermore, Reason describes "latent failures" that exist at the organizational layer, but he explicitly states that societal influences on organizational behavior (e.g., national culture, public perception of risk, regulation) are not controllable by managers and thus not relevant to his model. More recently, Leveson (2011) has proposed the System-Theoretic Accident Model and Process (STAMP), which more effectively captures the notion of feedback across the different layers of an organization and, at least implicitly, includes effects of societal influences that extend beyond the boundaries of the organization. In its effort to model interactions effects across levels, however, STAMP necessarily represents feedback as an abstraction based on the principles of control theory and thus does not directly incorporate the contributions of the diverse array of disciplinary methods described above. In real-world organizations, practitioners need an approach that builds on the systems orientation of STAMP while still providing concrete tools and methods relevant to the challenges that they face and firmly grounded in case studies of actual events. While Reason's point that managers have limited influence on societal factors is somewhat valid given our current limitations in this specific knowledge domain, the nature of the interactions among those factors must be included in a truly holistic approach to the problem, especially as a means to expand our knowledge of this area.

The proposed research program is focused on the development of an integrated methodology that is simultaneously theory-based, data-driven, holistic, and usable in real-world organizations. Just as the aviation industry has implemented checklists to ensure safety in the cockpit and as Pronovost has adapted that thinking to patient care (Pronovost and Vohr, 2010), the proposed work essentially will provide a checklist (of sorts) that can be implemented by systems engineers, leaders, and policymakers to ensure safety in complex systems. In specific contexts like aviation and healthcare, the successful use of checklists has required fundamental changes to the system to make their use both easy and palatable. Similarly, the goal of this research is not simply to define a set of procedural steps but rather to provide the knowledge and tools needed to adopt a new safety-oriented way of working across systems and organizations. The next section describes the approach to achieving the objectives of this ambitious research program.

2.3. RESEARCH APPROACH

The research effort is divided into three phases: Concept Incubation, Framework Specification and Database Development, and Supporting Studies on Theoretical Foundations. Phase 1 of the effort, lasting from February 24 to July 30, 2015, has already been completed, and this document represents the product of that effort. Phase 2 is the core of the project and an essential foundation for future research targeting particular aspects of the multilevel framework of system safety. This phase will address Objectives 1 and 2 of the overall program: (1) to develop a methodology for ensuring system safety that offers specific guidelines for system development, management, and accident investigation

by examining systems from multiple perspectives and (2) to develop a System Safety Database that will document methods and frameworks, case studies across several disparate contexts, expert contributions from several relevant fields, and the roles of various regulatory agencies and organizations in system safety. Phase 3 will address Objective 3 of the research by defining a series of interrelated but separable research projects focused on the theoretical foundations of each level of the framework. This phase represents a "modular" approach to the research program that facilitates a high degree of flexibility in scope, timing, and required support. While the most benefit will be derived from completion of all of these studies, any subset of them can be pursued either in parallel with or after completion of Phase 2. The remainder of this section describes the research program in detail. Sections 2.3.1 - 2.3.3 describe the three phases, and section 2.3.4 explains the qualifications of the research team to successfully complete the project.

2.3.1. PHASE 1: FOUNDATIONAL RESEARCH COMPLETED DURING CONCEPT INCUBATION

During the concept incubation period, the researchers completed a comprehensive review of an interdisciplinary and multi-domain body of literature on system safety, created an annotated bibliography based on the literature review, conducted a series of semi-structured interviews with noted experts (n = 16) on various aspects of system safety from across disciplines (e.g., engineering, human factors, organizational behavior, political science) and system contexts (aerospace, aviation, maritime, healthcare, petrochemical, nuclear), and developed a preliminary Excel version of the basic structure of the System Safety Database (called the System Safety Catalog at this stage in its development). With a team of seven graduate and undergraduate students, the literature was divided into six areas representing the relationships across the four levels of the framework (technical, human, organizational, and societal). Each semi-structured interview was recorded in digital audio and transcribed by team members. The interviews were used to obtain expert guidance on the literature review in each area and to solicit direct suggestions on the structure of the framework. In addition, the team analyzed each interview transcript using HyperResearch™ version 3.7.2. Content analysis was conducted using two methods, thematic analysis with *a priori* codes and grounded theory using emergent codes (Corbin and Strauss, 2014). Axial coding was then used to categorize themes resulting from the content analysis. Relative frequencies were used to assign weights to codes for later use in quantized models to test hypotheses about patterns across the data. The literature review and the results of the interview data analysis were used to develop the framework and to create the preliminary structure of the System Safety Catalog. These results will serve as the foundation for the core of the research program to be conducted during Phase 2.

2.3.2. PHASE 2: FRAMEWORK SPECIFICATION AND DATABASE DEVELOPMENT

The goal of this phase, which addresses Objectives 1 and 2 of the research, is to develop a holistic framework that integrates insights from across disciplines and industry contexts to provide a systematic tool for understanding system safety that will be useful to a wide array of practitioners, researchers, leaders, and policymakers. The System Safety Database will serve as the primary data source and the engine for operationalizing the framework. As mentioned above, the first phase of the research resulted in the development of a multilevel framework of system safety. This framework, shown in Figure 2.1, consists of 12 "lenses" organized into the four levels. Each lens represents an essential perspective that must be included in a truly holistic approach to system safety. The lenses, which have been defined through a rigorous process of literature review, expert interviews, and analysis are:

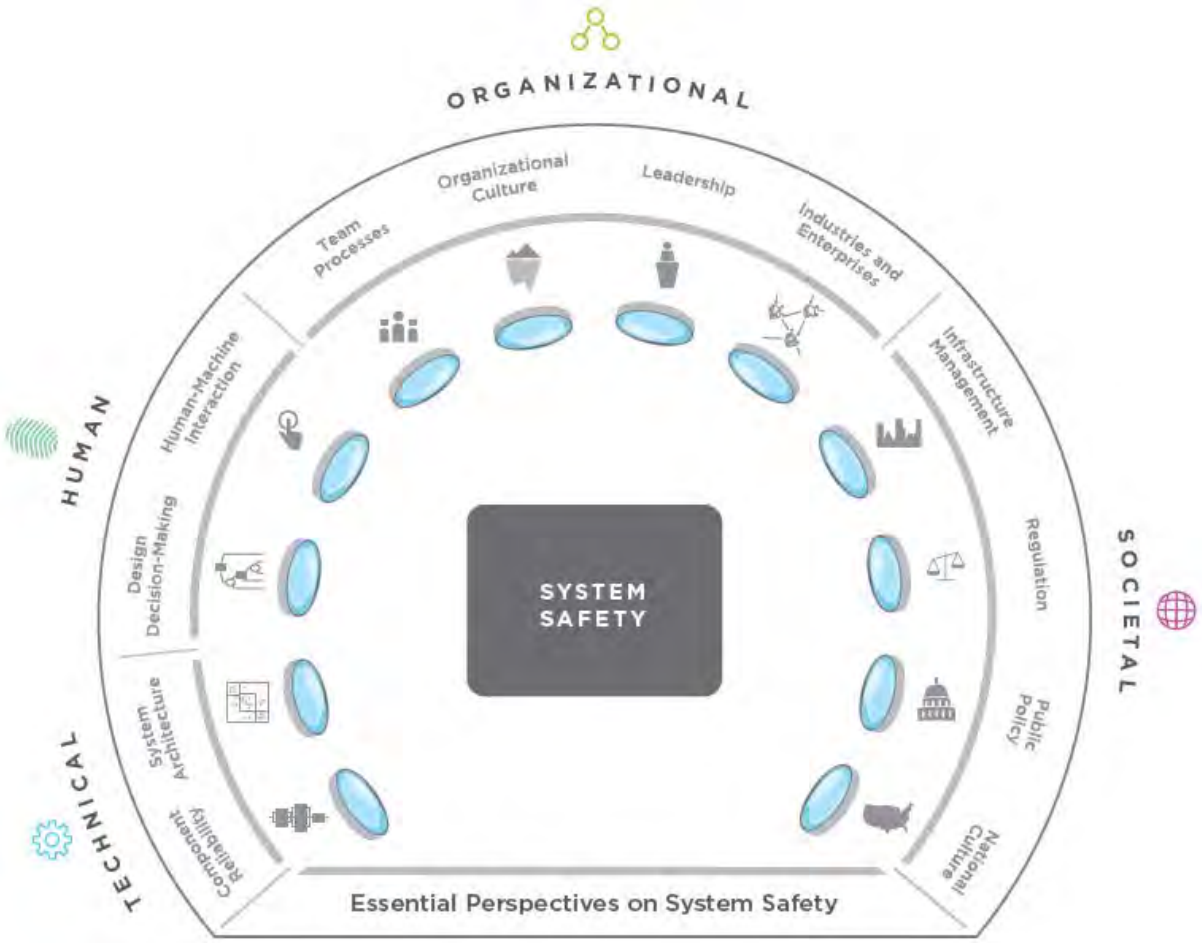


Figure 2.1. Multilevel Framework of System Safety. The framework consists of 4 levels and 12 "lenses" needed to fully assess safety in complex socio-technical systems from a holistic perspective.

- ***Component Reliability:*** Traditional quantitative study of safety using probabilistic risk models
- ***System Architecture:*** Upfront focus on ensuring safety in the design of engineered systems
- ***Design Decision-Making:*** Application of human factors and decision theory to human designers
- ***Human-Machine Interaction:*** Human factors focus on designing systems for human users
- ***Team Processes:*** Examination of information flow and shared knowledge in teams
- ***Organizational Culture:*** Study of safety culture/climate and its impact on safety outcomes
- ***Leadership:*** Role of leaders in shaping an organization's safety culture/climate
- ***Industries and Enterprises:*** Diffusion of safe practices across industries and supply chains
- ***Infrastructure Management:*** Challenges in ensuring safety in large-scale critical infrastructures
- ***Regulation:*** Influence of the legal and regulatory environment on system safety
- ***Public Policy:*** Role of government in ensuring safety and managing public perception of risk
- ***National Culture:*** Cultural influence on the other 11 lenses

The multilevel framework is a holistic taxonomy and a practical tool intended to guide the selection of methods and approaches for design, management, and research to ensure safer systems and more comprehensive accident investigations that lead to truly system-wide and lasting improvements. The

framework follows the logic of the use of checklists to ensure safety within particular systems but broadens the approach to provide, in essence, not checklists for individual aircraft pilots, healthcare providers, or other practitioners but rather a so-called "checklist of checklists" that will enable proper attention to all aspects of system safety. Just as the checklists used by healthcare providers to insert central lines are effective only if a "central line cart" with all necessarily supplies and equipment for the job are readily available and always stocked, the framework will be effective as the basis for a system safety checklist only if it is paired with the data, methodologies, and lessons learned needed to implement the items in the framework. The System Safety Database will serve as the repository of this knowledge and thus will be an essential part of the overall approach. This database will be capable of generating reports and providing essential information to anyone responsible for safety, investigating an accident, conducting research on system safety, and/or managing a hazardous system.

The first year of Phase 2 will involve analysis of accident investigation documents and semi-structured interviews with practitioners working at all levels and applying all lenses of the multilevel framework. Whereas Phase 1 involved only a traditional review of the literature, Phase 2 will extend that review and will include systematic grounded theory analysis of primary documents (e.g., memos, policy documents, testimonies) and quantitative textual analysis of secondary documents (e.g., accident investigations, panel transcripts) using Bayesian methods, following on a general approach advocated by Hopkins (2006). This type of analysis will lead to a more systematic understanding of contributing factors to accidents in complex systems. Furthermore, the inclusion of primary documents (e.g., original memos) will facilitate analysis both of catastrophic accidents that actually occurred and of events that did not necessarily result in major disasters but that could have been much more problematic under different circumstances. In another extension of the Phase 1 methodology, the research will once again involve expert interviews, but in Phase 2, the interviews will be with practitioners across all relevant levels and will focus on the key knowledge and equipment that they need to effectively maintain safety within their job functions. With 12 lenses in the framework and at least 4-6 interviews per lens, this phase of the work will involve several dozen interviews. Based on analysis of historical documents and grounded theory analysis of interview transcripts, a set of guidelines for maintaining safety by applying each lens will be created. This exercise will be repeated for each of the 12 lenses and across multiple system contexts. These guidelines will be formulated as system safety "checklists," as described above, and web-based surveys will be widely distributed to several hundred professionals across industries and job functions. The surveys will consist of questions aimed at assessing the validity of the checklist items and modifying the guidelines accordingly.

As stated above, the guidelines alone cannot be expected to lead to improvements in safety practices and outcomes. Appropriate tools and resources also must be made available to support their use (analogous to the central line carts used with Pronovost's checklists). For the guidelines of the multilevel framework, these tools and resources will be provided by the System Safety Database, which is intended to give the framework its "teeth" by providing the information backbone to enable implementation of the framework-derived guidelines. Currently, at the conclusion of Phase 1, the database exists in an early form as an Excel spreadsheet (the System Safety Catalog), consisting of four tabs: Methods and Tools, Case Studies, Agencies and Organizations, and Noted Expert Contributions. During the second year of Phase 2, one to two full-time database developers will be hired to build the System Safety Database as a comprehensive object-oriented database capable of generating reports, statistics, and lessons learned from a variety of disparate system contexts. Unlike some existing tools that document a subset of major accidents that have occurred within specific contexts, the System Safety Database will include a complete array of case studies, methods and tools, regulations and regulatory agencies, and expert contributions that will allow users to draw parallels and develop

appropriate strategies through systematic analysis of a complete set of case study data. While the hired programmers build the database architecture, the research team will conduct an exhaustive review of not only the academic literature but also accident investigations, news reports, press releases, and de-classified documents, including some that will require release through the Freedom of Information Act (FoIA). The resulting database tool, combined with the holistic framework, promises to provide a robust methodology for ensuring safety in the world's complex socio-technical systems. In the final year of Phase 2, the complete approach, including the framework, guidelines, and database, will be deployed in a pilot study with a selected company, and usability studies will be conducted to assess the approach.

2.3.3. PHASE 3: SUPPORTING STUDIES TO DEVELOP THE THEORETICAL FOUNDATIONS OF THE FRAMEWORK

The optional Phase 3 of the multilevel framework project represents a unique opportunity for the SERC. The research team is already developing a series of interdisciplinary research studies intended to more fully develop the theoretical underpinnings of each of the lenses of the multilevel framework. If the SERC effort ends after Phase 2, that investment promises to yield returns not only from achieving Objectives 1 and 2 but also from providing the intellectual platform for Objective 3. Thus, that work will have far-reaching impact beyond the immediate project since many of the efforts of Phase 3 will happen through other means. Moreover, by supporting one or more of the Phase 3 studies either in parallel with or following the Phase 2 investigation, the SERC will be able to ensure that a preferred portfolio of Phase 3 projects are conducted. In this way, this research program presents an opportunity to shape the priorities of the work to align with the strategic priorities of the SERC and its sponsors. The Phase 3 projects currently planned or underway include:

- ***Organizational Basis for Inherently Safer Design*** *(Lens: System Architecture).* The PI is currently in early discussions on a potential research effort building on the "mirroring" hypothesis (MacCormack, Baldwin, and Rusnak, 2012) to explore the role of structural changes in systems engineering organizations that can enable design for safety in complex engineered systems.
- ***Human Factors Analysis of Human-Model Interaction*** *(Lens: Design Decision-Making).* In a prior SERC technical report, Rhodes, Ross, Grogan, and de Weck (2015) propose an approach for analyzing "cognitive and perceptual considerations in human-model interaction." This presents an ideal opportunity to leverage an existing SERC project by conducting a human factors study on design decision-making. A traditional human-machine interaction study will be formulated but modified to involve human use of a system model during a simulated engineering design exercise. The goal of this simulation-based study will be to assess cognitive limitations of engineers that affect decisions regarding implementation of inherently safer design (Kletz, 2003).
- ***Cybersociophysical Systems*** *(Lens: Human-Machine Interaction).* The Co-PI is engaged in ongoing NIH- and NSF-funded human factors research with collaborators at the University of Virginia and the Carilion Center for Healthy Aging to use a complex human-machine system to predict environmental, psychosocial, and physical factors that interact to trigger agitation in persons with dementia.
- ***Information Flow and Shared Knowledge in Emergency Response Teams*** *(Lens: Team Processes).* The PI is currently developing a research study at the Emergency Operations Training Center (EOTC), operated by the Texas A&M Engineering Extension Service (TEEX), one of the world's leading facilities for emergency response training. This project will build on prior work of Avnet and Weigel (2013) by studying shared knowledge networks and team interactions in emergency response during simulated exercises at the EOTC.
- ***Network Analysis of Safety Culture*** *(Lens: Organizational Culture).* In this ongoing project (Avnet, 2015), the PI is working on the development of quantitative metrics for safety culture and

organizational learning based on the structure and evolution of shared knowledge networks in real-world organizations.

- ***Diffusion of CLABSI Prevention Protocols Across the Healthcare System*** *(Lens: Industries and Enterprises).* The PI has a pending proposal with the Agency for Healthcare Research and Quality (AHRQ) to build agent-based simulations for modeling diffusion of innovations in patient safety in an effort to identify network interventions that can improve the widespread use of known practices to reduce central line infections across the U.S. healthcare system.
- ***System Dynamics Modeling of Safety Regulations*** *(Lens: Regulation).* The research team is in the early stages of formulating a series of system dynamics studies to understand the regulation lens of the multilevel framework by modeling influencing factors in regulatory compliance to understand the unintended consequences that can result from certain regulations.
- ***Safety Climate and National Culture in Residential Construction*** *(Lens: National Culture).* The Co-PI has a pending proposal with the National Institute for Occupational Safety and Health (NIOSH) applying cultural ergonomics (Chapanis, 1974; Smith-Jackson and Essuman-Johnson, 2013) to residential construction as a means to identify relationships between cultural meta-schemas and system interaction.

2.3.4. RESEARCH TEAM QUALIFICATIONS TO ENSURE PROJECT SUCCESS

PI Avnet and Co-PI Smith-Jackson are uniquely suited for this work. Both have formal training and interdisciplinary education in system safety. The investigators are also complementary in their areas of expertise and bring strengths and peer connections across multiple areas. Between them, they have significant formal training, practical experience, and/or research experience that spans all 12 lenses of the multilevel framework.

Avnet has extensive research experience in system architecture, team processes, organizational culture, and public policy. In addition, he has worked as management consultant focusing on issues of performance improvement, culture change, and leadership, serving clients in a variety of private sector industries and in infrastructure management. He is the director of the System Architecture and Safety Management Laboratory, which conducts research on various aspects of system safety and on an array of topics directly related to aspects of system safety, including flexibility in system architecture, chronic healthcare management, and problems of infrastructure management related to the water-energy nexus. He is a faculty fellow of the Mary Kay O'Connor Process Safety Center (MKOPSC) and of the Center for Health Systems & Design (CHSD), and he is a member of the Texas A&M Systems Engineering Committee. He is the Texas A&M representative to the SERC and to the Council of Engineering Systems Universities (CESUN), and he has served as co-chair of the Systems Engineering track at the 2015 Industrial and Systems Engineering Research Conference (ISERC).

Smith-Jackson has conducted safety engineering research since 1996 with an emphasis on mixed methods, risk analysis, system safety methods and applications, and cultural factors in system safety. She is the director of the Human Factors Analytics Laboratory, which has strengths in modeling and analytics methods such as latent factor modeling. Her specific safety experience extends across various domains including agriculture, construction, public safety, forensics, and transportation. She has over 100 publications in the areas of safety, usability evaluation, and cultural ergonomics and has advised and co-advised 56 student dissertations and theses. One of her current projects examines usable, trustworthy, and persuasive technology principles applied to cybersociophysical systems. Her work in computer science includes a new project on applied identity modeling for cybersecurity.

The investigators continue to teach and advise student projects, theses, and dissertations in system safety and thus stay at the forefront of the field.

2.4. Relevance of the Research

The first mention of system safety in a formal document was in 1960s in Military Standard 882, now Military Standard 882-D: Standard Practice for System Safety (2012). At present, the military standard defines system safety as "the application of engineering and management principles, criteria, and techniques to achieve acceptable mishap risk, within the constraints of operational effectiveness and suitability, time, and cost throughout all phases of the system life cycle." While the definition places value on life cycles of systems, it focuses on only two disciplinary knowledge domains (engineering and management) and does not emphasize the need to address the complexities introduced by system interdependencies. In response to numerous catastrophic events over the years, system safety has evolved into an interdisciplinary area of study. Still, existing tools do not offer a truly holistic methodology that focuses both on interactions across multiple domains and the critical factors within each domain. This research will fill that gap by providing a concise yet holistic framework organized into a usable set of tools and driven by a powerful and comprehensive database. Just a decade ago, the time was not yet right for this type of approach to system safety, but by building on advances in systems theory and a growing trend toward interdisciplinary research, the proposed research offers a broad, practical perspective just when the appetite in the community for this type of approach has emerged.

The multilevel framework developed in this research program will lead to an actionable, data-driven, holistic approach to improve safety in socio-technical systems and to empower the system safety workforce of the 21st century. The framework, along with the accompanying guidelines and the System Safety Database, will provide tools and methods to be used by researchers across disciplines and by stakeholders at all levels, from individual operators to policymakers. The framework is important to researchers and practitioners in several distinct disciplines seeking to address safety challenges in complex socio-technical systems, and the lenses of the multilevel framework represent a concise list of the communities of practice that will benefit from the research. Furthermore, the principles of the framework are applicable to multiple types of systems and industry contexts. Indeed, the basic principles of the framework are likely to be applicable even beyond system safety. Although the framework was developed based on literature review and expert interviews on system safety, each of the identified lenses represents a set of issues that are important to any number of challenges in complex systems, including operational and financial performance, homeland security, and national defense. While the framework would need to be tested for its applicability to these other domains, it represents an important first step toward developing a common framework to address a wide variety of problems associated with complex socio-technical systems.

The research program is also highly relevant to the mission of the SERC and offers a specific contribution to each of the four SERC research areas. The research program contributes most directly to advancing the ***Trusted Systems*** area by complementing the programs on Systemic Security and Systemic Assurance with a new program focused on Systemic Safety. The research also contributes to the ***Enterprise Systems and System of Systems*** area through its focus on developing a framework to integrate a wide array of methods and tools. The research contributes to the ***Systems Engineering and Systems Management Transformation*** area in a more targeted way by contributing a specific methodology for testing the principles of human-model interaction, which forms an integral part of the Interactive Model-Centric Systems Engineering (IMCSE) program. Lastly, the System Safety Database is designed to

serve as an evolving body of knowledge on system safety, which helps to fulfill one of the basic goals of the ***Human Capital Development*** area. In addition, many of the specific projects outlined in section 2.3.3 in support of Objective 3 of the overall research program are aligned with various aspects of the four SERC research areas. Thus, this research program offers a unique opportunity to advance many of the priorities of the SERC and its sponsors.

2.5. EXPECTED CONTRIBUTIONS

The overarching contribution of this research is a truly holistic and integrative systems-oriented approach to safety in complex socio-technical systems. Whereas most existing approaches either focus on specific disciplinary approaches or provide a means for system-level analysis that is still fundamentally restricted to the methodologies of a particular discipline, this research aims to provide a means to examine system safety using a truly interdisciplinary approach that not only focuses on interactions across disciplines but also explicitly integrates the contributions of each of those approaches. Some of the more specific contributions correspond directly to the three main objectives of the research. First, the research will provide a robust framework that offers specific and practical guidelines to ensure system safety. Second, the work will include the development of a universal repository of safety-related knowledge that will help to ensure appropriate use of relevant tools and application of lessons learned when new hazards to complex systems emerge. Third, this work will provide a platform for further research on the theoretical underpinnings of each of the disciplinary perspectives described in the framework. Furthermore, the basic approach of applying lenses that represent several distinct disciplinary perspectives is generalizable to a host of other challenges in complex systems related to financial performance, system assurance, and many other issues. In addition, the framework will serve as a valuable educational tool for training interdisciplinary safety professionals. More broadly, it also will provide a platform for preparing the systems engineering workforce of the 21st century to address critical problems related to a broad range of technical, organizational, and political factors. In sum, this research is not only expected to contribute to the way that engineers, leaders, and policymakers approach system safety, but it also is intended to provide a more holistic approach to developing, managing, and providing assurance to a wide range of complex socio-technical systems.

2.6. RISKS AND MITIGATION STRATEGIES

There are several strategies built into the project to mitigate risks associated with timelines, achieving deliverables, and managing workload and resources. ***Risk:*** The project may be scoped too broadly to be completed in time. ***Mitigation Strategy:*** The work plan consists of a structured timeline with clear interim deliverables, biweekly virtual meetings, and quarterly in-person meetings to ensure that sufficient progress is made toward the plan. In addition, a formal evaluation team is included in the project budget. ***Risk:*** The scale of the database tool is highly ambitious. ***Mitigation Strategy:*** Full-time database programmers will be hired. ***Risk:*** The multilevel framework may not prove to be viable, or an appropriate pilot site may not be identified. ***Mitigation Strategy:*** The database and interdisciplinary analysis represent a minimal deliverable of the research program and a valuable resource to engineers, managers, policymakers, researchers. ***Risk:*** The potential exists for the research community to reject the basic premise of the multilevel framework. ***Mitigation Strategy:*** The research involves early inclusion of system safety experts through interviews and further downstream involvement through usability surveys, workshops, and panels. ***Risk:*** Resources may not available to complete all Phase 3 projects. ***Mitigation Strategy:*** The impact of Phase 2 is in establishing a platform for Phase 3 research, so actual Phase 3 projects can be scoped as a strategically aligned portfolio. In fact, it is not expected that all Phase 3 projects will be funded directly through this effort. While it is preferred that a subset of

Phase 3 projects be supported to ensure the ability of the SERC and its sponsors to determine the strategic direction of the overall program, much of the benefit that stems from Phases 1 and 2 of the program is the added value that will be derived from Phase 3 projects that will be supported by other sources based on the foundation provided by SERC investment in the development of the multilevel framework and the System Safety Database.

2.7. BUDGET

Although precise figures have yet to be calculated, the current estimated budget is $350K for Phase 2 and a total of $630K for a comprehensive Phase 3 program consisting of three to four projects focused on developing the theoretical foundations of selected lenses in the multilevel framework. Thus, the total budget for the overall research program, including the $20K already spent in Phase 1, is approximately $1 million.

2.8. TIMELINE

Figure 2.2 summarizes the project timeline. Phase 1 of this effort was completed during the incubation project period. Phase 2 consists of further specification and validation of the multilevel framework, development and testing of the System Safety Database, and pilot testing of the overall system, including the framework, the guidelines, and the database. In addition, overall project deliverables will be finalized during the third year of Phase 2. At present, the research has led to the identification of a large volume of supporting research studies developing the theoretical foundations of the framework, and the number of these studies will continue to expand as Phase 2 progresses. The optional Phase 3 will consist of one or more of these supporting studies pending the outcomes of Phase 2 of the research and the strategic priorities of the SERC and its sponsors.

Research Phase	Task Description	2015			2016				2017				2018		
		Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3	Q4	Q1	Q2	Q3
1	*Concept Incubation* (Define Framework Based on Literature Review and Expert Interviews)														
2	*Framework Specification* (Conduct/Analyze Practitioner Interviews, Collect/Analyze Primary and Secondary Documents)														
	Framework Validation (Create Guidelines, Distribute/Analyze Surveys, Iterate Framework and Guidelines)														
	Database Development and Testing (Define Database Structures, Categorize System Safety Data, Test Database)														
	System Pilot Test (Integrate Framework/Database Package, Install System in Pilot Organization, Conduct Usability Studies)														
3	*Optional Supporting Studies on Theoretical Foundations* (Select Portfolio of Projects Based on Strategic Priorities)			*Optional: Can Begin During or After Completion of Phase 2 Work*											
2,3	*Project Evaluation* (Formative and Summative Evaluations)														
2,3	*Dissemination* (Publish in Peer-Reviewed Journals, Present at Conferences, Organize Workshops and Panel Discussions)														

Figure 2.2. Project Timeline.

2.9. PROJECT EVALUATION AND MEASUREMENT

This research is scoped to include the use outcomes-based evaluation using a logic model, as shown in Table 2.2. Inputs include the resources and information necessary for project implementation, including information gained from the initial concept incubation period. An implementation model will be developed at the beginning of the project. Outcomes and outputs describe the deliverables that are part of the initial project and the forthcoming larger effort. The evaluation method consists of two tiers. Tier 1 includes formative assessments to be conducted on an annual basis, and Tier 2 is a summative

evaluation to be completed during the last quarter of the project timeline. The external evaluator will use the full logic model as the basis for evaluation. The following success metrics will be used for formative assessment and summative evaluation. For formative assessment (Tier 1), experts will review the outcomes and outputs at the end of each year of the project and will use a set of quality metrics to provide ratings. For summative evaluation (Tier 2), the same metrics will be used to assess the overall completion of the project against stated goals. Metrics that are likely to be used in these evaluations include expert ratings of database quality, effectiveness, and potential impact; effectiveness of dissemination of results (by monitoring website visits, publication downloads, workshop attendance); and community of practice surveys to measure framework usefulness and validity. Evaluation constructs that map to the "success metrics" in Table 2.2 include ratings of "holism" of the multilevel framework; framework transparency, practicality, and generalizability; usability of the System Safety Database; comprehensiveness of database content; face validity of the methodology; content validity of the methodology; downloads of the tool; and blogs and other public comments on the multilevel framework and the database.

Table 2.2. Logic model to structure formative and summative project evaluation

Logic Model for Project Evaluation*			
INPUTS	**ACTIVITIES**	**OUTPUTS/OUTCOMES**	**IMPACTS**
• PI/Co-PI • Resources • Implementation Model • System Safety Experts • Research Assistants • Programmers • Literature	• Archival Analysis • Interviews • Qualitative and Quantitative Modeling • Guideline Formulation • Database Development	• Multilevel Framework • System Safety Database • Platform for Targeted Research • Convergence Among System Safety Community of Practice	• Holistic Approach Useful to Industry and Government • Codification of Safety Knowledge • Modifications to Existing Models • Regulatory and Policy Impact • Public Awareness

* Scope of evaluation contingent on level of support

REFERENCES

Abras, C., Maloney-Krichmar, D., and Preece, J. (2004). "User-Centered Design." In Bainbridge, W. (Ed.), *Encyclopedia of Human-Computer Interaction*. Thousand Oaks, CA: Sage Publications, 37(4), 445-456.

Apostolakis, G. E. (2004). "How Useful is Quantitative Risk Assessment?" *Risk Analysis* 24(3): 515-520.

Ashford, N. A. (2007). "The Legacy of the Precautionary Principle in US Law: The Rise of Cost-Benefit Analysis and Risk Assessment As Undermining Factors in Health, Safety and Environmental Protection." In Bainbridge, W. (Ed.), *Implementing the Precautionary Principle: Approaches from the Nordic Countries, the EU and the United States*. London, UK: Earthscan.

Avnet, M.S. (2015). "A Network-Based Approach to Organizational Culture and Learning in System Safety." *Procedia Computer Science* 44: 588-598. (CSER 2015: Conference on Systems Engineering Research, Hoboken, NJ, 18-19 March 2015.)

Avnet, M.S. and Weigel, A.L. (2013). "The Structural Approach to Shared Knowledge: An Application to Engineering Design Teams." *Human Factors: The Journal of the Human Factors and Ergonomics Society* 55(3): 581-594.

Baybutt, P. (2003). "On the Ability of Process Hazard Analysis to Identify Accidents." *Process Safety Progress* 22(3): 191-194.

Carayon, P., Schoofs Hundt, A., Karsh, B.-T., Gurses, A.P., Alvarado, C.J., Smith, M., and Flatley Brennan, P. (2006). "Work System Design for Patient Safety: The SEIPS Model." *Quality and Safety in Health Care* 15(Suppl 1): i50-i58.

Casler, J.G. (2014). "Revisiting NASA as a High Reliability Organization." *Public Organization Review* 14(2): 229-244.

Chapanis, A. (1974). "National and Cultural Variables in Ergonomics." *Ergonomics* 17: 153-175.

Columbia Accident Investigation Board. (2003). *Report, Volume 1*. Washington, DC: National Aeronautics and Space Administration and Government Printing Office.

Cooper, G.E. (Ed.). (1980). *Resource Management on the Flight Deck*, Vol. 2120. Mountain View, CA: Ames Research Center, National Aeronautics and Space Administration.

Corbin, J. and Strauss, A. (2014). *Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory*, 4th ed. Thousand Oaks, CA: Sage.

Department of Defense. (2012). *Standard Practice MIL-STD-882E*. Washington, DC: Department of Defense.

Donabedian, A. (2005). "Evaluating the Quality of Medical Care." *Milbank Quarterly* 83(4): 691-729.

Dunjó, J., Fthenakis, V., Vílchez, J. A., and Arnaldos, J. (2010). "Hazard and Operability (HAZOP) Analysis: A Literature Review." *Journal of Hazardous Materials* 173(1): 19-32.

Dyer, M.K., D.G. Little, E.G., Hoard, A.C., Taylor, and Campbell, R. (1972). *Applicability of NASA Contract Quality Management and Failure Mode Effect Analysis Procedures to the USFS Outer Continental Shelf Oil and Gas Lease Management Program* (PDF). National Aeronautics and Space Administration George C. Marshall Space Flight Center. TM X–2567.

Furuya, E.Y., Dick, A., Perencevich, E.N., Pogorzelska, M., Goldman, D., and Stone, P.W. (2011). "Central Line Bundle Implementation in US Intensive Care Units and Impact on Bloodstream Infections." *PLoS One* 6(1): e15452, 6 pages.

Hixenbaugh, A. F. (1968). "Fault Tree for Safety." *No. D6-53604*. Seattle, WA: Boeing Company, Support Systems Engineering.

Hopkins, A. (2006). "Studying Organisational Cultures and Their Effects on Safety." *Safety Science* 44(10): 875-889.

Kennedy, R. and Kirwan, B. (1998). "Development of a Hazard and Operability-Based Method for Identifying Safety Management Vulnerabilities in High Risk Systems." *Safety Science* 30(3): 249-274.

Kletz, T. A. (1997). "Hazop – Past and Future." *Reliability Engineering & System Safety* 55(3): 263-266.

Kletz, T. (2003). "Inherently Safer Design – Its Scope and Future." *Process Safety and Environmental Protection* 81(6): 401-405.

Kohn, L.T., Corrigan, J.M., and Donaldson, M.S. (Eds.). (2000). *To Err Is Human: Building a Safer Health System.* Washington, DC: National Academies Press.

Lee, W.S., Grosh, D.L., Tillman, F.A., and Lie, C.H. (1985). "Fault Tree Analysis, Methods, and Applications: A Review." *IEEE Transactions on Reliability* 34(3), 194-203.

Leveson, N.G. (2011). *Engineering a Safer World: Systems Thinking Applied to Safety*. Cambridge, MA: MIT Press.

Levine, M.E., & Forrence, J.L. (1990). "Regulatory Capture, Public Interest, and the Public Agenda: Toward a Synthesis." *Journal of Law, Economics, and Organization* 6(special issue): 167-198.

Liang, S.Y. and Marschall, J. (2011). "Commentary: Update on Emerging Infections: News From the Centers for Disease Control and Prevention." *Annals of Emergency Medicine* 58(5): 450-451.

MacCormack, A., Baldwin, C., and Rusnak, J. (2012). "Exploring the Duality Between Product and Organizational Architectures: A Test of the 'Mirroring' Hypothesis." *Research Policy* 41: 1309-1324.

Morgan, M.G. (1993). “Risk Analysis and Management.” *Scientific American* 269: 32-41.

Norman, D.A. and Draper, S.W. (Eds.). (1986). *User Centered System Design: New Perspectives on Human-Computer Interaction*, 1st ed. Hillsdale, NJ: Lawrence Erlbaum.

Occupational Safety and Health Administration. (2000). *Process Safety Management, OSHA 3132, Reprinted*. Washington, DC: U.S. Department of Labor.

Payne, S.C., Bergman, M.E., Beus, J.M., Rodríguez, J.M., and Henning, J.B. (2009). “Safety Climate: Leading or Lagging Indicator of Safety Outcomes?” *Journal of Loss Prevention in the Process Industries* 22(6): 735-739.

Perrow, C. (1994). “The Limits of Safety: The Enhancement of a Theory of Accidents.” *Journal of Contingencies and Crisis Management* 2(4): 212-220.

Perrow (2011). “Fukushima and the Inevitability of Accidents.” *Bulletin of the Atomic Scientists* 67(6): 44-52.

Presidential Commission on the Space Shuttle Challenger. (1986). *Report to the President*. Washington, DC: Government Printing Office.

Pronovost, P., and Vohr, E. (2010). *Safe Patients, Smart Hospitals: How One Doctor's Checklist Can Help Us Change Health Care from the Inside Out*. New York, NY: Plume.

Rasmussen, J. (1983). “Skills, Rules, and Knowledge; Signals, Signs, and Symbols, and Other Distinctions in Human Performance Models.” *IEEE Transactions on Systems, Man and Cybernetics* 3: 257-266.

Rausand, M. (2004). “Preliminary Hazard Analysis.” Presentation. Trondheim, Norway: Norwegian University of Science and Technology.

Reason, J. (1990). “The Contribution of Latent Human Failures to the Breakdown of Complex Systems.” *Philosophical Transactions of the Royal Society B*, 327: 475-484.

Reason, J. (1997). *Managing the Risks of Organizational Accidents*. Aldershot, UK: Ashgate.

Rhodes, D., Ross, A., Grogan, P., and de Weck, O. (2015). “Interactive Model-Centric Systems Engineering (IMCSE). SERC Phase Two Technical Report.” *SERC-2015-TR-048-2*. Hoboken, NJ: Systems Engineering Research Center.

Roberts, K.H. (1990). “Managing High Reliability Organizations.” *California Management Review* 32(4): 101-113.

Shappell, S.A. and Wiegmann, D.A. (2001). “Applying Reason: The Human Factors Analysis and Classification System (HFACS).” *Human Factors and Aerospace Safety* 1(1): 59-86.

Smith-Jackson, T. and Essuman-Johnson, A. (2013). “Cultural Ergonomics: Overview and Methodologies.” In Smith-Jackson, T.L., Resnick, M., and Johnson, K. (Eds.), *Cultural Ergonomics: Theory, Methods, and Applications*. Boca Raton, FL: CRC Press.

Stigler, G.J. (1971). “The Theory of Economic Regulation.” *The Bell Journal of Economics and Management Science* 3-21.

Vicente, K.J. (1999). *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work*. Mahwah, NJ: Lawrence Erlbaum.

Weick, K.E., Sutcliffe, K.M., and Obstfeld, D. (2008). “Organizing for High Reliability: Processes of Collective Mindfulness.” *Crisis Management* 3: 81-123.

Wu, A.W., Lipshutz, A.K., and Pronovost, P.J. (2008). “Effectiveness and Efficiency of Root Cause Analysis in Medicine.” *Journal of the American Medical Association* 299(6): 685-687.

Zohar, D. (2010). “Thirty Years of Safety Climate Research: Reflections and Future Directions.” *Accident Analysis & Prevention* 42(5): 1517-1522.

3 DETECTING AND EVALUATING TECHNICAL DEBT OF SOFTWARE SYSTEMS (LIGUO HUANG, SMU)

3.1 INTRODUCTION

The purpose of this project is to define a set of methods, processes, and tools (MPTs) to detect and evaluate levels of technical debt (TD) in existing software and to apply this information to affordability decision activities related to:

- Single software system maintenance and retirement
- Evolution and enhancement of system of systems (SoS) that rely on software-intensive systems

The ultimate goal is to identify MPTs that can be used to quantify system TD so that it can be used to refine cost and schedule estimates related to the evolution/modernization of existing systems, many of which are part of one or more SoS.

3.1.1 BACKGROUND: CURRENT STATE OF SOFTWARE SYSTEM DEVELOPMENT

The demand for software systems increases by 900% each decade. Although expenditure on software development increases by 200% each decade, productivity of software system engineers only increases by 35% (Leon, 2004). Approximately 75% of the total life cycle cost for a given system is related to system maintenance and operation (Jorgensen and Molokken-Ostvold, 2006; Gilb, 2006; Standish Group, 2014a; Standish Group, 2014b; Cerpa and Verner, 2009; Linberg, 1999; Chen and Huang, 2009; Tang and Huang, 2011). Technical debt has been recognized as a critical factor that leads to the escalating maintenance cost of software systems. Technical debt (TD) refers to delayed technical work or rework that is incurred when short-cuts are taken or short-term needs are given precedence over the long-term objectives. If the debt is not paid off, then it will keep on accumulating interest, making it hard to implement changes later on. Unaddressed technical debt increases software entropy. Gartner estimates that the Global "IT Debt" has the potential to grow to $1 trillion by the end of 2015 (Gartner, 2014). CAST Research Lab (CRL) analyzed 1,400 applications containing 550 million lines of code (LOC) submitted by 160 organizations and estimated that the Technical Debt (TD) of an average-sized application of 300 KLOC is $1,083K, which represents an average TD per LOC of $3.61. TD has become a major issue leading to the budget overrun of many software projects.

3.1.2 GENERAL CAUSES AND IMPACTS OF TD

It has been observed that the increasing complexity of design, code and final system along with system evolution is a major cause of TD (Leon, 2004; Gery et al., 2014), which eventually results in a software system that is difficult or even impossible to maintain due to the architecture/design erosion. A good example in relation to TD is the recently launched U.S. healthcare.gov system. The overall complexity of the system (more than 5000 pages in the Affordable Care Act and more than 78,000 qualified health plans across the federally serviced states) and the need for simplicity in the user experience were a significant challenge. Within the 5000 pages of the act, the probability of ambiguous language and conflicting requirements was extremely high. While concurrency is fundamental to good development and necessary to roll out a large system in a relatively short time frame, it requires adequate planning and the development of a solid foundation to be successful. It appears that careful system architecting, requisite management infrastructure to allow rapid and efficient supply chain communications, and engineering process adaptability were not adequately in place before engaging 55 individual contractors to perform more than $600 million in federally contracted work (Boehm et al., 2014). A careful analysis of the document and agreement among stakeholders on the interpretation of requirements before committing to architecture, schedules, and contracting seems critical. However, top-level government

leadership in the U.S. Department of Health and Human Services (HHS) had no significant experience in overseeing a systems development effort anywhere close to the size and complexity of healthcare.gov and therefore little understanding of commitments as well as shortfalls in coordination among the constituent system development. Newly published information indicates that the White House and HHS leaders were informed in a March 2013 independent assessment from McKinsey & Co., that the "launch was fraught with risks (Boehm and Turner, 2003)." Some of the key problems documented in the report (unstable requirements, little time for testing, and little to no time for fixing problems) were never resolved before the rollout (Boehm et al., 2014).

If TD is additive for separate, decoupled systems, it will become a multiplicative impact for strongly coupled SoS as experienced in the development of the large scale healthcare.gov system. Some example sources of multiplicative TD for strongly coupled SoS include but not limited to:

- Shortfalls in one component system likely to violate assumptions in other SoS components;
- Fixes in one component system may degrade other SoS components;
- Component systems often have weak visibility into content of other SoS components;
- Except for Directed SoSs, no single manager can guide cross-component fixes;
- Fixes more time-consuming, less likely to succeed.

Future system development trends will exacerbate the problem of TD proliferation especially in SoS Engineering (SoSE):

- Trend toward Internets of Things actually software-intensive
- Millions of apps and components to choose from
- Increasing need for interoperable systems: coordinated defense, crisis management, coordinated services
- Proliferation of independently-developed infrastructure services
- Continuing priority of time-to-market over software assurance.

3.2 Motivations, Objectives and Impacts

3.2.1 Limits of Current Research and Practice

Technical debt (TD) is multifaceted. Common TD elements include system defects, unimplemented features, incomplete tasks, poor/obsolete artifacts, or unmaintainable/un-evolvable software systems due to poor architecture/internal organization of software. Since Ward Cunningham first drew the comparison between technical complexity and debt in his experience report in 1992 (Cunningham, 1992), research on TD detection and evaluation has been prosperous and fruitful in multiple domains including software architecture/design and software code quality in both academia and industry. Over the past three decades, about a hundred of studies have been published, with their topics ranging from TD conceptual analysis (e.g., (Siebra et al., 2012; Schmid, 2013), detection (e.g., (Marinescu, 2004; Wong et al., 2011; Marinescu, 2012; Zazworka et al., 2014), to evaluation (e.g., (Izurieta et al., 2013; Ktata and Lévesque, 2010; Nugroho et al., 2011). Although the increasing number of studies produced significant benefits in defining and assessing TD as well as improving software quality (Sharma, 2012; Ramasubbu and Kemerer, 2013; Griffith et al., 2014), *there lacks of a consistent and consolidated view of the definitions and determinant factors of TD, which may result in confusion on its detection and evaluation in both academic research and industrial practice*. For instance, in the state-of-the-art TD research, code smell is often considered as a cause of TD (Fontana et al., 2012; Guo et al., 2010; Ligu et al., 2013) that could be naturally classified as the code debt. However, the God Class, as a type of code smell, is occasionally used to indicate the design debt (e.g., (Griffith et al., 2014; Zazworka et al., 2011). In

industrial practice, Klinger et al. (Klinger et al., 2011) addressed *the lack of effective ways for stakeholders with different kinds of software quality concerns to communicate and reason about TD*. Various studies have proposed diversified views on the nature (metrics and models) of TD, which dampens the utilization of TD as a conceptual and technical gauge of systems and software maintenance and evolution cost.

During software system development and evolution, the knowledge of TD levels can help inform decisions as to whether one should continue to evolve a given software system or replace it with a new one (since it may be cheaper to develop a new system than to fix all of the TD and then evolve the existing system). Likewise, if TD is limited to a few areas (components), it may be more cost effective to refactor those areas to eliminate the TD, and continue to evolve the system. In SoSE, understanding the level of technical debt in a component or constituent subsystem will help decide which component (or subsystem) to include and which to replace, retire, or exclude. *However, the diversified and even inconsistent understanding of TD measurement imposes additional barriers for project success-critical stakeholders (e.g., project managers, architects and developers) to efficiently select the appropriate TD metrics from a large pool of candidates.* Codabux and Williams (Codabux and Williams, 2013) reported that *developers tend to use their own taxonomy of TD based on the development tasks they were assigned as well as their own interpretation of the measurement terms and criteria*. Meanwhile, *various stakeholders may have their own value propositions in defining and evaluating TD*. For instance, project managers may be concerned about the overall project or development team rework cost and the lack of knowledge transmission as knowledge distribution debt that could compromise the quality of an evolving software product (Tom et al., 2013), while developers may focus on the extra effort incurred by defect fixing. Consequently, different metrics have been proposed and employed to measure TD based on various perspectives. Thus it is important to understand the overlaps in TD definitions and measurement to help engineering teams improve their TD assessment as well as approaches for detecting and resolving TD.

Despite the multiplicative impact of TD on the rework cost in developing and maintaining System of Systems (SoS), *very few state-of-the-art TD research and practice have addressed the issues of TD in System of Systems Engineering (SoSE).*

3.2.2 OBJECTIVES

Being aware of the aforementioned limits of current technical debt (TD) research and practice, the **top-level objective of our proposed research is to structure, select and extend the existing software system TD analysis methods, processes, and tools (MPTs) to best support System of Systems (SoS) TD detection and remediation activities as well as to understand the cost/schedule impacts if there is no (or limited) remediation**. The specific SoS TD MPTs to be further investigated include:

- evolving ontology of TD concepts and metrics which constructs a catalogue of TD categories with the metrics that measure the extent/impact of the TD and the estimated cost to remediate the TD if any, as part of the software quality ontology development;
- automated TD evaluation with respect to current and potential constituent systems to augment the risk assessment and mitigation in SoSE;
- integration of TD assessment component into the next generation system and software cost/schedule estimation models (COCOMO and COSYSMO) to support the SoS development and evolution affordability analysis (cost vs. value).

3.2.3 IMPACTS

The proposed SoS TD evaluation MPTs as well as the empirical evidences collected, synthesized and analyzed in this work can significantly improve the current TD and software quality research and practice by (1) developing a consolidated and integrative ontology which provides a multi-perspective visualization and traceability of the TD categories, definitions, metrics with assessment methods and tools; (2) systemizing the documented TD detection/evaluation metrics and experiences into a comprehensive while structured knowledge base for researchers and practitioners to select the best fit for their project decision making scenarios; (3) enabling automated TD evaluation at various granularities of SoS capability development, integration and evolution; (4) augmenting the next generation system and software cost/schedule/quality estimation models with an integrative TD assessment component.

Who cares? The SoS project decision makers, system and software engineers (including SoS architects and developers) who are affected by or concerned with the quality and rework cost of SoS development and evolution at U.S. Department of Defense (DoD), SERC, DoD-affiliated industrial organizations will benefit from this research.

3.3 APPROACH

3.3.1 OVERVIEW

We have planned an end-to-end set of research tasks to meet the current and future challenges of TD described in Section 3.1 as it impacts SoS capability development, integration, evolution and associated affordability decisions. Figure 3.1 presents the workflow of our proposed five research tasks. During the incubation period, we have completed a comprehensive and in-depth systematic literature review on software system technical debt detection and evaluation (Task 1) together with a complementary online survey with the participation of both academic researchers and industrial practitioners (Task 2). The empirical data collected from Tasks 1 and 2 have established a solid foundation for the subsequent post incubation Tasks 3-5. We are now ready to develop the rich and evolutionary TD ontology to streamline, structure and relate its concepts and measurement (Task 3).

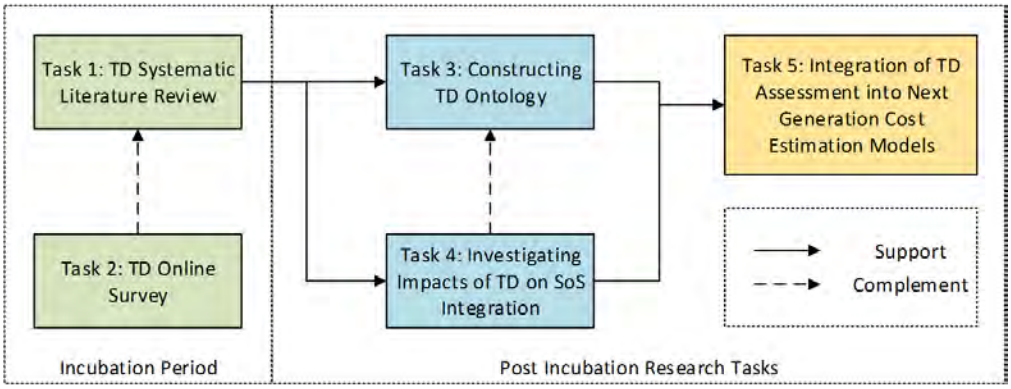


Figure 3.1: An overview of SoS TD evaluation research tasks

TD emerging during SoS integration not only compromises the quality of SoS but also incurs an increasing amount of rework cost during SoS integration, maintenance and evolution. Due to the multiplicative TD on strongly coupled SoS as mentioned in Section 3.1, we will explore the MPTs for

evaluating the impacts of single system TD on SoS Integration, as well as identify the critical areas of SoS interoperability assessment (Task 4). Then we will augment the TD ontology with the new concepts, methods and tools for evaluating the SoS Integration Debt. Further, in collaboration with the University of Southern California (USC), we will integrate the TD assessment component into the next generation software system cost/quality estimation models used to estimate SoS capability development effort and schedule. Each task will be elaborated in the following sections.

3.3.2 TASK 1: TECHNICAL DEBT (TD) SYSTEMATIC LITERATURE REVIEW (SLR)

To set up a foundation for developing the ontology of TD concepts and metrics, we have conducted a comprehensive systematic literature review (SLR) following Kitchenham's methodological guidelines (Kitchenham and Charters, 2007) to streamline and structure the state-of-the-art TD detection and evaluation methods, metrics and tools. Figure 3.2 shows the eight major steps of the review process. Figure 3.3 shows our search strategy, an integrated (Zhang et al., 2011) two-step approach, which includes (1) an initial manual search on domain-specific venues (i.e., International Workshop on Managing Technical Debt) and generic venues with high reputations in Software Engineering (i.e., ICSE, ASE, FSE, OOPSLA, ICSME, ESEM, EASE, FASE) and (2) an automated search across 8 digital libraries using refined search strings constructed with the keywords identified from the manual search. To the best of our knowledge, no other empirical study on TD has been published with the same depth of the research questions and breadth of the scope as this review.

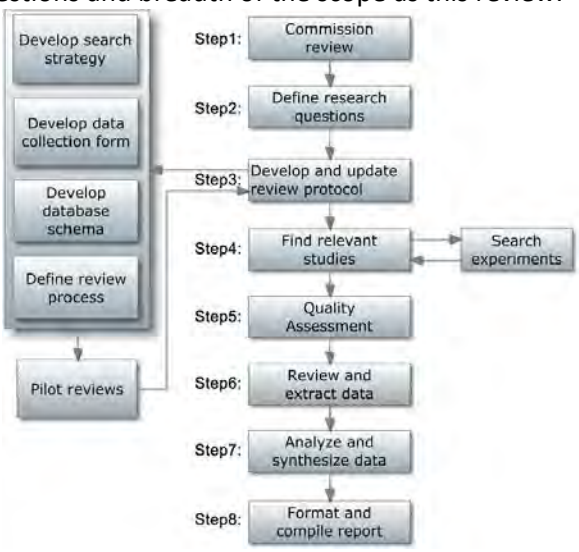


Figure 3.2: Review process flowchart

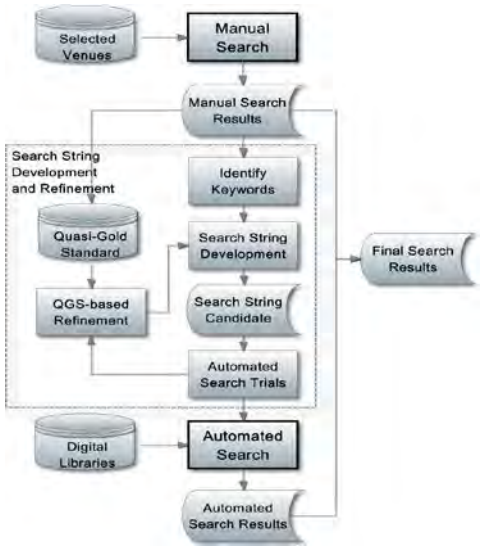


Figure 3.3: The two-step search approach

A total of 86 papers published between 1992 and 2014 in software engineering venues were retrieved and included in our review where the empirical evidences were collected to answer the following five research questions.

- ***RQ1.*** *How do people define technical debt and what are the major root-causes of technical debt reported by the included studies?*
- ***RQ2.*** *How can we categorize TD? What categories of TD have been reported? Has each category of TD been clearly defined in the included studies?*
- ***RQ3.*** *What kinds of metrics have been used for evaluating TD in software systems?*
- ***RQ4.*** *How can we fit TD management into the software development life cycle (SDLC)? What are the unique characteristics of TD with respect to the agile development?*
- ***RQ5.*** *What is the strength of empirical evidence presented in the included TD studies?*

Section 3.4.1 will summary our main findings and answers to these research questions.

3.3.3 TASK 2: TECHNICAL DEBT ONLINE SURVEY

As a complementary empirical study to the TD SLR described in Section 3.3.2, we have hosted an online survey on the SurveyMonkey[1] to solicit expert opinions of both academic research and industrial practice on TD detection and evaluation. The data collected from the survey also validate certain conclusions drawn from the SLR. The TD online survey questionnaire includes not only 10 questions from the TD domain but also 6 questions from the software refactoring domain. Software refactoring research has been found overlapped with TD research, especially on maintenance debt, architecture debt, etc. However, our TD SLR excluded most of the software refactoring papers due to the study selection criteria which requires the included studies specifically contain the metaphor term "debt". Thus the software refactoring related questions were asked in the survey.

The survey started in November 2014 and ended in March 2015. We received 37 completed responses from both academic researchers and industrial practitioners. Figure 3.4 shows the demographic distribution of the respondents. A majority of the respondents (48.8%) have been conducted academic research for over 10 years. The industry experience is evenly distributed over less than 1 year to more than 10 years. The top five research areas of the respondents include software architecture (38.5%), software measurement (38.5%), software design (33.3%), software quality management (30.8%), and software development (25.6%) respectively (excluding "others", 30.8%). A respondent could be involved in more than one research areas. Section 3.4.2 will discuss a few interesting findings from the online survey.

Figure 3.4: Demographic analysis of TD online survey

3.3.4 TASK 3: CONSTRUCTING TECHNICAL DEBT ONTOLOGY

One of the major contributions of the TD SLR is to have collated a systematic taxonomy of metrics that measures the principle and interest for different TD categories. Although the taxonomy provides an unambiguous hierarchy between TD categories and quantitative metrics, it is still necessary to create a systematic formal specification that captures the interrelationships among concepts, metrics, methods

[1] Survey Link: https://www.surveymonkey.com/r/MY2RP7J

and tools for TD evaluation. Hence, we propose to develop a TD ontology that encompasses a catalogue of TD categories, definition, metrics, and assessment methods/tools as well as their relationships.

Based on the Ontology Development 101 (Noy et al., 2001), our goals of developing the TD ontology are:

- To share common understanding of TD knowledge among researchers and practitioners;
- To support the effective selection of the "best fit" TD metrics for the SoSE affordability analysis;
- To enable the reuse of TD knowledge (categories, definitions and metrics) and make the TD knowledge application assumptions explicit;
- To support the evolution of TD knowledge.

To meet these goals, we will develop the TD ontology along with a Technical Debt Ontology Sharing and Evolution (TDOSE) method and tool. The TDOSE tool is a web-based system designed for TD ontology evolution as shown in Figure 3.5. TDOSE has two major capabilities: (1) visualization of TD ontology for knowledge sharing; and (2) an editor of TD ontology specifications which enables continuous evolution of the ontology.

The TD ontology evolution process involves three different roles: (1) general user, (2) evolution contributor, and (3) evolution committee member, each granted a different level of access to the TD ontology repository.

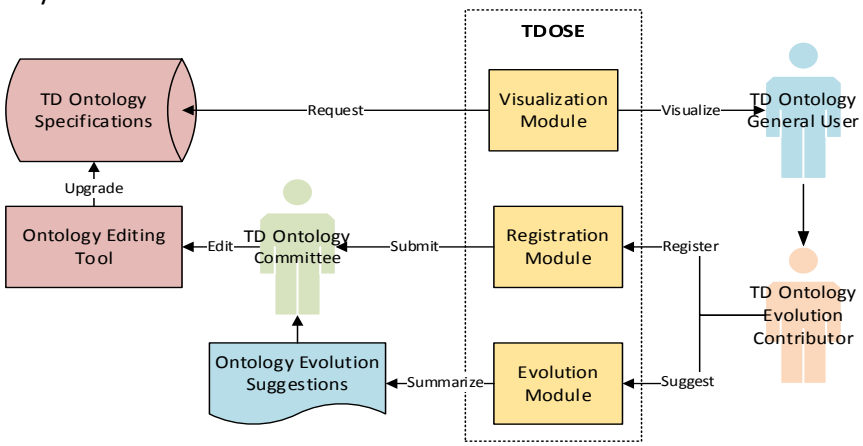


Figure 3.5: Evolutionary process of TD ontology

During the initial stage of TD ontology development, an open-source ontology editor, Protégé, will be used for editing and visualizing TD ontology. However, we will develop our own web-based TDOSE to replace Protégé in support of the online iterative evolution of the ontology. We plan to build TDOSE on emerging technologies, such as WebGL[2] to support a web-based 3D graph rendering. We will evaluate the formal ontology specification languages including OWL (Web Ontology Language), SHOE (Simple HTML Ontology Extensions), XOL (Ontology Exchange Language), etc.

To demonstrate how TD ontology will facilitate the TD evaluation in research and practice, Section 3.4.3 will go through an illustrative example of how TDOSE can be used for TD ontology visualization and knowledge sharing.

[2] WebGL (Web Graphics Library), more information at https://www.khronos.org/webgl/

3.3.5 TASK 4: INVESTIGATING IMPACTS OF TD ON SOS INTEGRATION

TD evaluation and management for SoS integration, maintenance and evolution is overlooked by the state-of-the-art TD research. Our TD SLR (Section 3.3.1) reveals that only 2 out of 86 studies emphasized the importance to manage TD in integrated code/components/systems produced by different vendors that is carried over through either internal or external supply chain during system acquisition (McGregor et al., 2012; Monteith and McGregor, 2013). However, the supply chain debt (as described in these two studies) is insufficient to measure the impacts of TD from the SoSE perspective. Thus this research task will investigate the impacts of TD on SoSE in three aspects:

- TD of constituent systems that impacts SoS integration;
- TD of connectors (glue code, middleware, etc.) between constituent systems that impacts SoS evolution;
- TD incurred by insufficient interoperability assessment due to project resource constraints that may result in the suboptimal decision of system selection.

TD in a constituent system. *Unsolved known or latent TD carried over from a constituent system could eventually impose multiplicative impacts on a SoS and incur extra rework when it is reused and integrated with other systems in the SoS.* For example, system code debt that essentially affects the code comprehension would increase the complexity of the code configuration and/or tailoring during system integration. TD in a single system can be detected and evaluated using existing TD assessment MPTs if one can access the system internal artifacts (e.g., source code, documentation, etc.), using the current TD assessment methods. For instance, if a constituent system module must be tailored for the integration with other systems, modular dependency analysis methods (e.g., Design Structure Matrix (DSM)) are able to tell whether and how much extra maintenance effort is needed to make the required changes due to architecture debt (Wong et al., 2011). Nevertheless, not all TD categories would impact the SoS integration. Hence, **we propose to conduct an empirical study to investigate what types of TD could cause extra efforts during SoS integration and how much extra effort they incur.** We will determine the subset of TD measurement models and metrics that are (1) mature enough and (2) can support an assessment of TD with respect to extra rework cost estimation during SoS integration. The taxonomy of TD metrics constructed from our SLR (Task 1) and the TD ontology establish a solid foundation to perform the empirical evaluation of TD measurement methods for different TD types. We will synthesize the TD principal and interest measurements and examine the impacts of different types of TD on SoS integration effort. The impacts are measured as the extra rework effort incurred in reusing and maintaining the constituent systems during SoS integration. We will follow the following steps to conduct the empirical study. The workflow is shown on the left of Figure 3.6.

Step1	Extend the TD SLR (Task 1) to include the refactoring studies.
Step2	Investigate what types of TD would impact SoS integration efforts and select TD measurement models
Step3	Define the TD adjustment factor(s) to be incorporated into the cost estimation model

TD in connectors. In addition to the TD carried over from constituent systems, TD could also occur in the development/configuration of connectors (glue code, middleware, etc.) between systems. It can be incurred by either the initial suboptimal design choice or the future architecture/design decay of the SoS integration connectors. Such TD will incur the increasing amount of rework effort during SoS evolution when upgrades or changes are needed for constituent systems that are integrated by connectors with TD. We are able to leverage or extend the existing MPTs (e.g., Wong et al., 2011, Zhang et al., 2015b) for detecting and evaluating the design/code debt to examine the TD in SoS integration connectors.

TD due to insufficient SoS interoperability assessment. During SoS capability development, if several systems with similar capabilities available for integration, a sub-optimal decision might be made in selecting the constituent systems because of insufficient SoS interoperability assessment usually due to project resource constraints including schedule, budget, personnel skills, etc. If the complete SoS interoperability assessment is infeasible under limited resources, some integration decisions should have more priority than others. Where shall the SoS interoperability assessment focus on? To answer this question, **we propose a prioritization method based on (1) the system dependency analysis using Design Structure Matrix (DSM), and (2) a prioritized checklist of system interaction inconsistency analysis by system analyzer to identify the critical areas of SoS integration decision and interoperability assessment.** Lane and Valerdi have started their research on SoS interoperability assessment by synthesizing the existing SoS interoperability models and examining their impact on SoSE effort (Lane and Valerdi, 2011). Our research will be a complement to their interoperability assessment work. The prioritization workflow is illustrated on the right of Figure 3.6. The step-wise prioritization method is detailed in the table below.

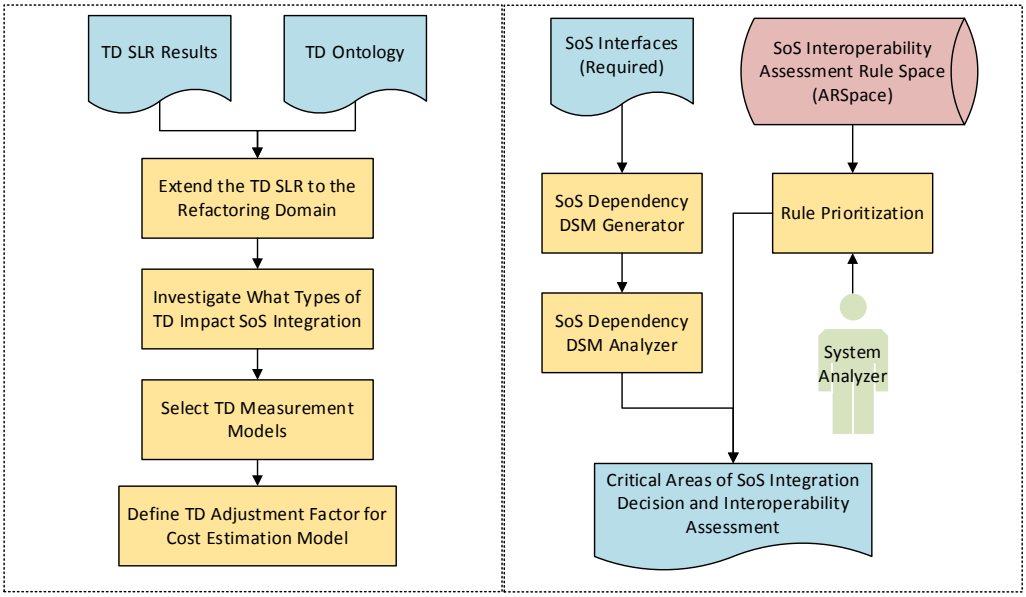


Figure 3.6: Investigating impacts of TD on SoS integration

Step 1	Take the inputs of the interfaces (e.g., APIs) of constituent systems.
Step 2	Perform system dependency analysis via the SoS Integration Design Structure Matrix (DSM) to generate system dependency impact ratings.
Step 3	Create the SoS interoperability assessment rule space (ARSpace) by prioritizing rules by system analyzer. (Note that Step 2 and 3 can be done in parallel.)
Step 4	Compute interoperability assessment (IA) weights based on system dependency impact ratings from step 2 and prioritized ARSpace rules from step 3.
Step 5	Generate the prioritization suggestion on SoS interoperability assessment for the SoS integration decision maker.

Hence, Task 4 consists of three concurrent subtasks: (1) investigating how TD of constituent systems will impact the system maintenance, reuse, as well as SoS integration (led by SMU); (2) Prioritizing and identifying the critical areas of SoS interoperability assessment (led by SMU); and (3) SoS interoperability assessment (led by USC).

To demonstrate the feasibility of this approach, Section 3.4.4 will first describe a software system maintainability prediction method and tool (*SMPLearner*) which has been developed by the PI's team and can be leveraged to assess the maintainability debt in a single software system. Then we will walk through the SoS interoperability assessment prioritization with a pilot example.

3.3.6 TASK 5: INTEGRATION OF TD ASSESSMENT INTO NEXT GENERATION COST ESTIMATION MODELS

The detection and evaluation of TD will eventually support the SoS affordability decision making. To meet this goal, we plan to integrate the TD assessment result for each constituent system of SoS as the adjustment factor(s) into the cost estimation models to estimate the adjusted reuse effort of each system. This task will be accomplished in collaboration with the Center for Systems and Software Engineering (CSSE) at the University of Southern California (USC) in the following two steps.

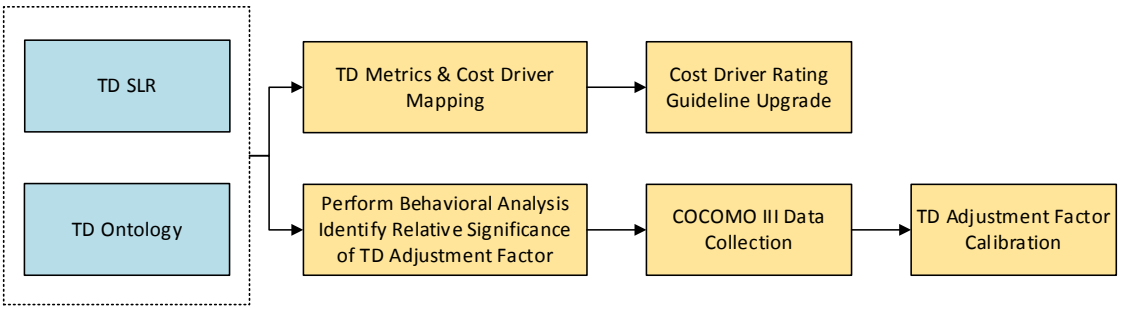


Figure 3.7: Integrating TD assessment into next generation cost estimation models

Step 1. Determine how to incorporate the TD principal and interest measurement of each constituent system into the next generation system and software cost estimation model COCOMO III, which will be an integration of COCOMO [3] (COnstructive COst MOdel) and COSYSMO [4] (Constructive Systems Engineering Cost Model). In this step, we will start from the maintenance and reuse models and parameters in COCOMO and COSYSMO. The reuse model of COCOMO II includes a number of factors that help estimate the effective software system size (ESLOC) by taking into account the percentage of design and code to be modified, percentage of system to be integrated, design and code quality of the current system, etc. The maintenance model of COCOMO II helps the estimation of effort during system maintenance which is defined as the process of modifying existing software system while not changing its primary functions (Boehm et al., 2000). Maintenance during SoS integration includes redesign and recoding of small portions of the original system, redesign and development of interfaces, and minor modification of the system structure. Maintenance can be classified as either updates or repairs. The maintenance effort estimation formula is the same as the COCOMO II Post-Architecture development model (with the exclusion of *SCED* and *RUSE* cost drivers and a different set of effort multipliers for *RELY* cost driver). The basic form of the COSYSMO model is an engineering size estimate raised to a calibrated exponent (to account for the diseconomy of scale as size grows (Valerdi et al., 2011)) multiplied by an effort adjustment factor (which is the product of a set of effort factors). To measure the impacts of carried-over debt from a constituent system on SoS integration effort, we will map both TD categories and their corresponding evaluation metrics into the sizing parameters and cost drivers in the reuse and maintenance models of COCOMO II and in COSYSMO. The mapping would guide us how to take the impacts of TD into consideration during the system/software cost estimation. As shown in Figure 3.7, the mapping will be built upon our TD SLR (Task 1) and TD ontology (Task 3) which provides an overall

[3] COCOMO II: http://csse.usc.edu/csse/research/COCOMOII/
[4] COSYSMO: http://cosysmo.mit.edu/

landscape of TD concepts, metrics, methods and tools. Once the mapping is completed, we will improve the rating guideline for the relevant sizing parameters and cost drivers in COCOMO and COSYSMO as well as incorporate the TD adjustment factor into the cost estimation model.

To better illustrate how TD assessment would be relevant to and help with the cost driver ratings, let us start with a realistic scenario in estimating the rework effort in software system maintenance (e.g., fixing defects, adding/retiring features, refactoring). Assuming the COCOMO II maintenance effort estimation model is used to estimate the future system maintenance effort PM_M (see Eq.1), the Low $RELY$ rating indicates the low reliability of the system which requires extra rework effort to fix latent faults.

$$PM_M = A \times (Size_M)^E \times \prod_{i=1}^{n} EM_i \quad \text{Eq. 1}$$

The God class is an important indicator of design debt which easily leads to latent faults in the system. Deferring the refactoring of God classes would usually impose a risk of massive extra future maintenance effort if dependencies are continuously built on it. The example metrics for detecting the design debt due to the existent of God classes are as follows.

Detection of God class for design debt measurement:		
Access To Foreign Data	ATFD > 5	Number of accesses to data in other classes (both directly or indirect)
Weighted Method Count	WMC > 46	Sum of McCabe's cyclomatic complexity of all methods
Tight Class Cohesion	TCC < 0.33	Internal cohesion of the class

The detection of many God classes in a system (or system module) implies a Low *RELY* rating. The current cost driver rating guideline did not specify how these TD metrics can be added to guide the rating. One of the benefits of leveraging these lower-granularity TD metrics is that they can be automatically extracted from system code or documents using static analysis tools such as *CodeVizard* (Zazworka and Ackermann, 2010).

Step 2. Calibrate the newly incorporated TD adjustment factor in COCOMO III. After defining the TD adjustment factor in COCOMO and COSYSMO, its data collection and calibration will be incorporated into COCOMO III. Additionally, the initial rating scales of the TD adjustment factor can be retrieved and/or derived from the empirical results reported by the TD studies that were included in our SLR (Task 1) and software system refactoring studies. We will follow the seven step modeling methodology applied in COCOMO II which minimizes the risk of consuming a lot of people's time and effort supplying data that produces poor estimates (Boehm et al., 2000).

3.4 Preliminary Results

This section presents our preliminary results to demonstrate the feasibility of our proposed approaches.

3.4.1 Task 1: Technical Debt (TD) Systematic Literature Review (SLR)

The results of TD SLR have established a consolidated and integrative view of technical debt organized around five research questions on TD definitions, root causes, categories, metrics, characteristics in SDLC processes, and strength of empirical evidence. The main findings of the SLR are as follows:

(1) Existing definitions of TD are divergent and inconsistent because they are usually tied with different root causes of TD in software systems.
(2) Although the architectural, design and code related root causes are prominent in the state-of-the-art TD research, the potential root causes of TD pertaining to requirements and process management are not well addressed in the TD research.

(3) We have identified 17 categories of TD and defined each of them. The top 5 TD categories that are prominent in the TD detection and evaluation reside in the process areas of design, documentation, architecture, defect, and code, while TD categories in other process areas are rarely addressed or well defined. Besides, the boundary between TD categories needs to be clarified for effective communication in collaborative development.
(4) The definitions of TD categories are usually aligned with their sources to enable the measurement of each category of TD. Obstacles in measurement, evaluation, or prediction of TD result in the fact that a majority of TD research prefer to analyze source code as the basis to of TD detection or and evaluate TD evaluation.
(5) The lack of quantitative TD interest metrics prevents the business manager from making the right decision on system development and evolution due to the inaccurate estimation of TD impacts.
(6) The evaluation of TD at different levels of granularity can be complementary with one another in that the low-level metrics can provide evidences to support or cross-validate the measurement with high-level metrics
(7) The detection and evaluation of TD along with its data collection should be considered as a long term software quality assurance and risk assessment activity that needs to be integrated and distributed into the entire software development life cycle (SDLC).
(8) The earlier manifestation of TD in agile development could be both a challenge and opportunity for the TD management. If an appropriate TD management detection and evaluation strategy is taken, we can detect and prevent the accumulation of TD principal and interest earlier in agile development than waterfall.

In addition to the above findings, **we have collated a taxonomy of metrics extracted from the SLR (see Appendix 3.A) that measure the TD principal and interest for different TD categories**. The taxonomy of TD metrics strives to provide a "best fit" approach for measuring a specific category of TD in practice, with which we hope to establish a systematic knowledge landscape for TD detection and evaluation so that knowledge, workload and obligations can be allocated to the right personnel in SDLC. This taxonomy can benefit from more extensive empirical studies. With better approaches and tools for quantifying TD, both principal and interest, system and software development decision-makers can make more informed decisions with respect to continuing system affordability, when to retire systems that have reached their end of life, as well as how to better utilize existing systems to develop new capabilities through the formation or enhancement of System of Systems (SoSs).

The complete TD SLR results and 86 included studies are detailed in our journal paper (Zhang et al., 2015a) submitted to the Journal of Systems and Software on May 31, 2015.

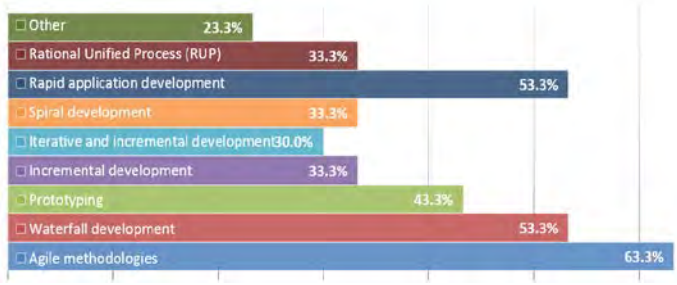
3.4.2 TASK 2: TECHNICAL DEBT ONLINE SURVEY

This section discusses the data collected from the respondents' answers to a few interesting survey questions, as shown in Figure 3.8.

What are the major sources of Technical Debt?

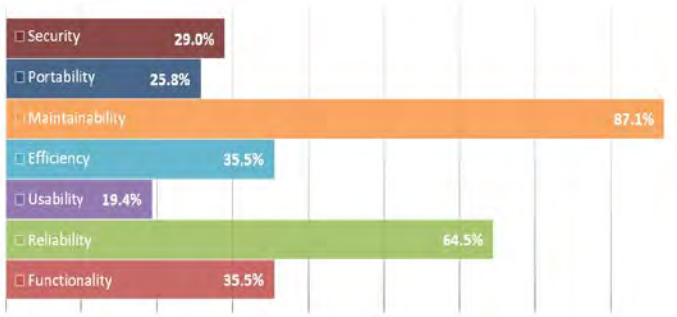
We have observed almost evenly distributed concerns about the major sources of TD from the respondents spanning over processes, architecture, design, code, defects, etc., which echoes our SLR findings and partially explains why the current TD definitions are diversified and inconsistent.

Which development process methodology is the most susceptible to Technical Debt?



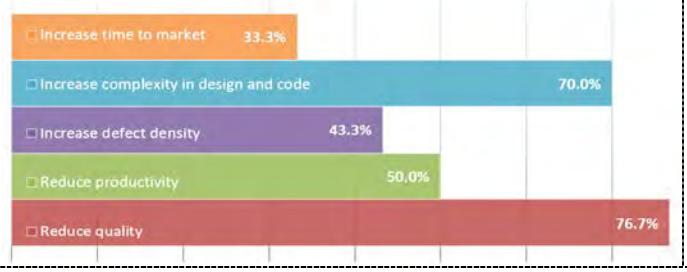
The agile and rapid application development processes are the top two processes strongly influenced by TD. This also concurs our finding in the TD SLR. Consequently, in SLR we have investigated how we can fit TD management into the software development life cycle (SDLC) as well as the unique characteristics of TD with respect to the agile development.

Which of the following quality attributes are highly influenced by Technical Debt?



Researchers and practioners agree that Maintainability is the top quality attribute that is highly influenced by TD. Along the line of this obeservation, we have developed an automated method and tool to learn software maintainability from the internal code quality metrics (Zhang et al., 2015b). Further, we will investigate how TD will affect the software system maintainability, SoS changeability/evolvability and system rework cost during the proposed research tasks 3, 4 and 5.

Deferred refactoring tasks due to the new features in related components, modules, or subsystems will:



The deferred refactoring tasks, which incurs TD, will eventually compromise the quality and increase the complexity of software system design and code.

Figure 3.8: Technical Debt Online Survey Results

3.4.3 Task 3: Constructing Technical Debt Ontology

We plan to develop the Technical Debt Ontology Sharing and Evolution (TDOSE) tool leveraging WebGL and HTML 5 technology to support interactive TD knowledge sharing, editing and evolution. This section walks you through an illustrative example to show how a partial TD ontology is visualized in TDOSE as depicted in Figure 3.9.

Four types of nodes in the hyperlink visualization represent the concepts at different hierarchies of TD ontology: (1) root node – TD, (2) TD category node, (3) TD detection/evaluation approach/tool node, and (4) TD detection/evaluation metrics node. The root node is Technical Debt (TD), the parent of all TD categories. The left bottom chart in Figure 3.9 shows the "root" centered view, where the interrelationships between different TD categories are also displayed. The interrelationships between TD categories will be initially captured based on the empirical evidence in *state-of-the-art* literatures (from Task 1). For instance, the documentation debt and people debt may be correlated because the empirical evidence showed that insufficient documentation would impact the knowledge distribution which aggravated the people issue (e.g., sliding developer productivity, personnel volatility).

The visualization module enables an interactive multi-perspective view of the ontology. One type of nodes can be linked to all other three types of nodes, which enables users to easily navigate the ontology to trace a type of TD to its metrics, detection/evaluation approaches and tools. Users can select a specific perspective (node) which becomes the center of the view with all other connected nodes (representing directly linked concepts) surrounding it. Clicking the "code debt" node invokes "code debt" centered view as shown in the upper right chart in Figure 3.9. The "code debt" is linked to all other three types of nodes. If a TD metric node (McCabe Complexity) is clicked, we may easily visualize this metric has been used for measuring different types of TD as well as in different TD detection/evaluation approaches/tools (as shown in the TD metrics centered view in the right bottom chart of Figure 3.9). At the same time, we are also able to detect a TD metric that has never been used by any approach/tool, which may lead to a further investigation why this metric has not been used in the state-of-the-art practice.

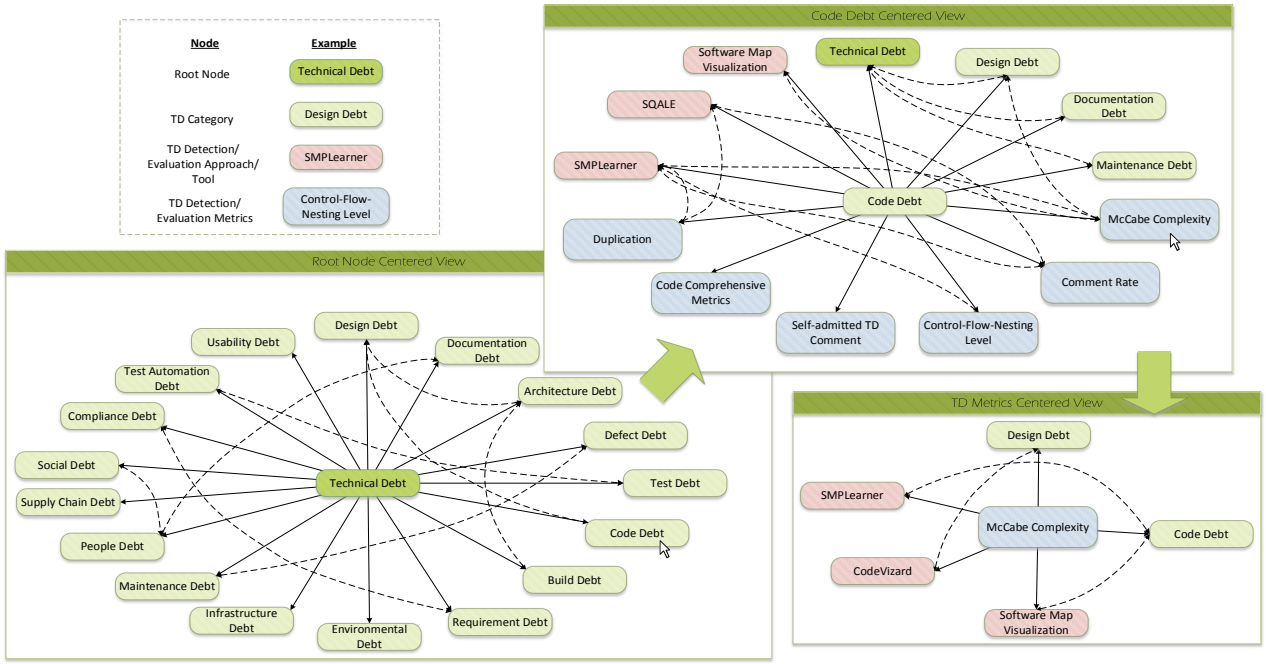


Figure 3.9: TD ontology visualization example

3.4.4 Task 4: Investigating Impacts of TD on SoS Integration

This section first presents the method and tool (*SMPLearner*) (Zhang et al., 2015b) that have been developed by the PI's team to automatically assess the maintainability (changeability) of a single

software system. We shall be able to harvest the existing TD measurement methods/tools and empirical results reported by literatures in both TD and software system refactoring domains to analyze the impacts various types of TD on SoS integration. Then we demonstrate the feasibility of the SoS TD impact analysis approach described in Section 3.3.5 via a pilot example on an open source SoS (Taiga).

Maintainability Debt Assessment for a Single Software System: *SMPLearner*. *SMPLearner* accepts two inputs: (1) application source code releases; (2) historical code change repository containing all the commits and the churned eLOC (sum of the added, deleted and changed lines of code; change of a line is counted twice) for each file of a release. It systematically evaluates and compares 24 supervised learning algorithms to learn the Software Maintainability Prediction Model from the code evolution history. Figure 3.10 depicts an overview of the *SMPLearner* approach which consists of two major components: (1) Metrics Collection, and (2) Machine Learner. The **Metrics Collection** component gathers two kinds of metrics to create training and testing corpus: 1) the actual maintenance effort measured by ME or total Churned eLOC, which is extracted from the code change history; and 2) the 4-level hierarchical code metrics collected from the source code releases. The **Machine Learner** component is developed in two steps: (1) learning and constructing a maintainability prediction model leveraging the best learning algorithm selected based on a comparison of accuracy of the maintainability predicted by models learned via 24 machine learning algorithms, and (2) analyzing 4-level metrics' impact on maintainability.

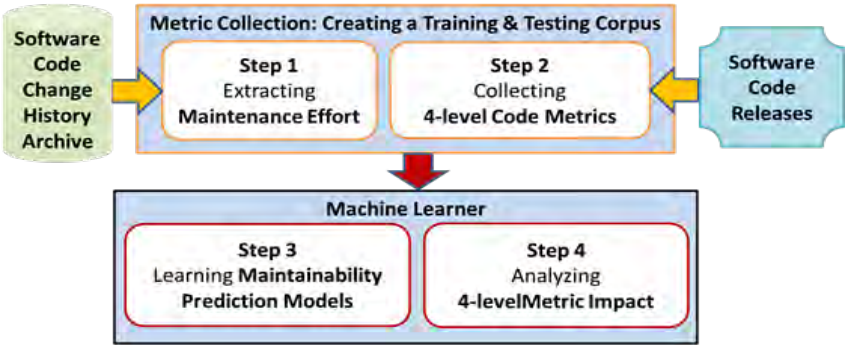


Figure 3.10: *SMPLearner* Overview

Specifically, *SMPLearner* makes the following contributions to the maintainability assessment for a single software system:

1. To make it clear what characteristic of maintainability our approach can be applied to. *SMPLearner* specifically contributes to the **prediction of software changeability sub-characteristic of maintainability by learning from the code evolution history**. Our measurement of changeability covers both corrective and adaptive/evolution changes.
2. To assess the actual maintainability, ***SMPLearner* novelly introduces the actual average maintenance effort (*ME*) derived based on the number of commits and the churned eLOC into the maintainability model learning and correlates it with a set of objective software code metrics collected by the static analysis tool**. Both the number of commits and the churned eLOC can be automatically extracted from the code change history. We consider the metric ME a better representation of actual maintainability than just the total churned eLOC used in recent studies (Kaur and Kaur, 2013; Malhotra and Chug, 2012; Van Koten and Gray, 2006; Zhou and Leung, 2007) because it takes into account the average effort taken by developer to change each line of code while the total churned eLOC only measures the total changed lines of code which may include both quick and effort consuming changes.

3. Instead of learning from a single level of code metrics (e.g., (Kaur and Kaur, 2013; Malhotra and Chug, 2012; Van Koten and Gray, 2006; Zhou and Leung, 2007), ***SMPLearner*** **learns a software maintainability model from a much richer set of 4-level hierarchical software code metrics including 44 metrics** in total. **Further experiments with *SMPLearner* reveal the relative contributions of the four code levels to maintainability prediction** on both metrics of code change effort: *ME* and total churned eLOC, respectively.
4. ***SMPLearner*** **has been experimented with 24 machine learning algorithms and systematically evaluated on a much larger data set of 8 open source systems with approximately 22.8 Million SLOC**. It significantly improves the accuracy of maintainability prediction when using *ME* as the metric of code change effort as well as a basis of evaluation, where accuracy is measured by the Spearman's Rank Correlation Coefficient (*SRCC*) between the predicted maintainability (*MP*) and converted maintainability from *ME*. Compared with the traditional *MI*-based model, the best learning algorithm of ***SMPLearner*** **improves the prediction accuracy by 5.19 times on the 8 large scale open source software systems**.

A pilot example of SoS interoperability assessment prioritization. Taiga (https://taiga.io/) is an open-source SoS with an integration of six open-source projects. It provides a project management platform for agile system developers. The level 1 architecture diagram of Taiga is shown on the left of Figure 3.11. Taiga consists of major systems (green blocks), internal systems (blue blocks), and external systems (yellow blocks). "Front" refers to the client and "Back" refers to the server. Both client and server are able to dynamically load external systems (e.g., Slack, a team communication solution provider). In addition, the API documents of the constituent systems and the glue code connecting these systems are also accessible at the Taiga's GitHub repository.

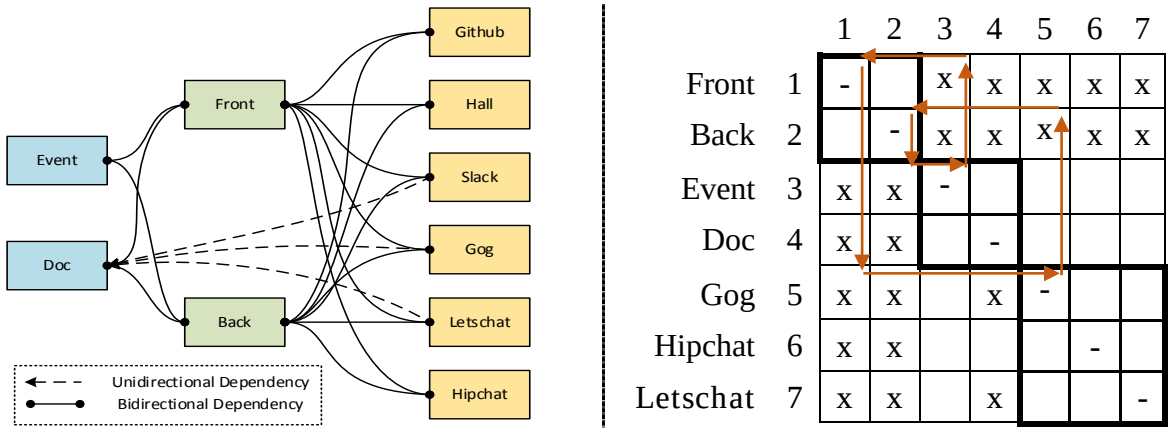
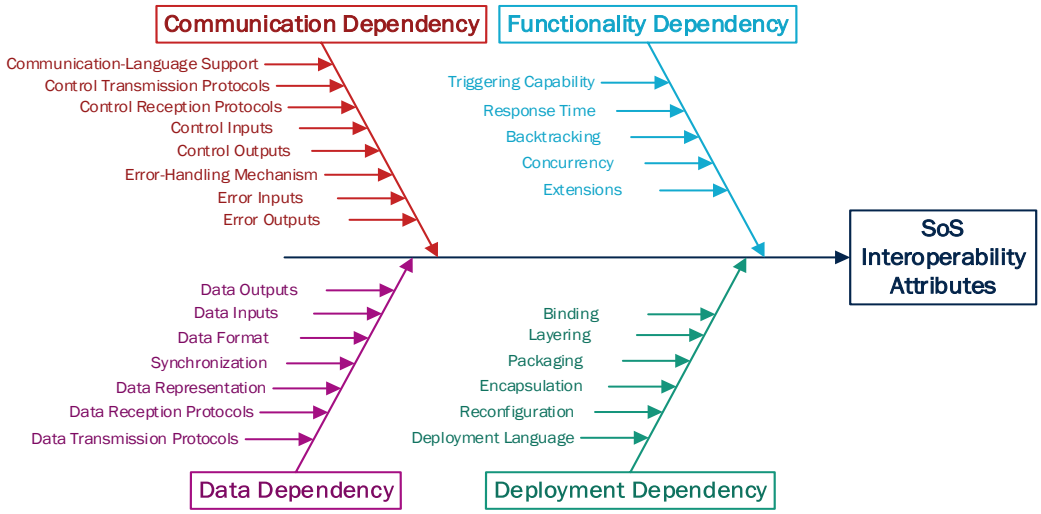


Figure 3.11: A partial DSM for Taiga system dependency analysis

DSM System Dependency Impact Analysis. A partial SoS dependency DSM of Taiga is shown on the right of Figure 3.11. In the matrix, each "X" represents a direct dependency where the row parameter (a system) is affected by the column parameter (a system). For instance, the "Front" system is affected by the "event" system and vice versa as shown on the left of Figure 3.11. The bolded blocks in the matrix enclose the dependency internal to a group of constituent systems while the "X" out of these blocks indicates the external dependencies across these system groups. We are able to detect the unidirectional/bidirectional, transitive and/or cyclical dependencies among constituent systems via DSM. The cyclical dependency among systems (as marked by the red arrows on the right of Figure 3.11) can be problematic for SoS designers, given changes to a system may propagate via a chain of dependencies to many other systems. In such a structure, the presence of cyclicality means that there is

no guarantee that the design process (or a design change) will converge on a globally acceptable solution that satisfies all constituent systems (Steward, 1981).

Prioritization of Interoperability Assessment Rules. In addition to the DSM system dependency impact analysis, certain interoperability assessment rules can be more important than others for a pair of systems to be integrated (represented by an "X" in DSM). In this illustrative example, we provide a sample checklist of interoperability assessment, where the interoperability assessment attributes are classified by four kinds of system dependencies (i.e., communication dependency, functionality dependency, data dependency, deployment dependency) as shown in Figure 3.12. Figure 3.12 also shows a few examples of interoperability assessment rules. Each rule indicates a specific kind of system interaction inconsistency. Missing the assessment of a relevant and critical rule will lead to future rework to resolve the inconsistency problem during SoS integration and evolution. The system analyzer suggests the priority of assessment rules based on her experience of how much effort it may take to resolve an inconsistency problem. The higher the potential rework effort, the higher priority an assessment rule is assigned.



- Data Connection Mismatch: A data connector utilized for interaction between two systems that are not compatible to them
- Triggering Actions Mismatch: Two systems support triggering but the sequence in which actions take place may be unclear and resulting in undesired effect
- Predictable Call Response Time Mismatch: (Triggered) Call from a system that requires predictable response times to another system not originally considered
- Error Handling Mismatch: Two communicating systems where at least one system does not support any error handling mechanism resulting in synchronization issues

Figure 3.12: Example SoS interoperability assessment attributes and rules

Note that the example interoperability assessment attributes and rules presented above will be expanded by more comprehensive SoS interoperability models (e.g., Information System Interoperability Model (LISI) (DOD, 1998)) USC will lead the effort of SoS interoperability assessment.

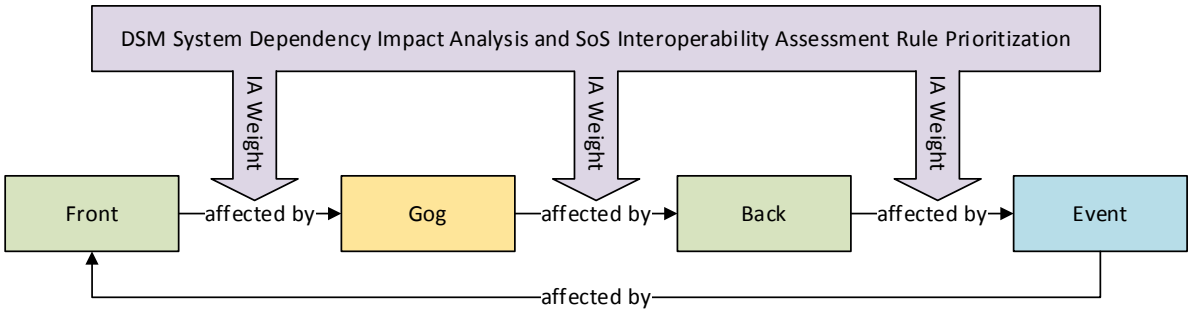


Figure 3.13: Applying SoS interoperability assessment priority in the SoS dependency chain

Interoperability Assessment (IA) Weights. Finally, the IA weight is determined by a combination of the system dependency impact rating from DSM system dependency analysis and the priority of each interoperability assessment rule assigned by system analyzer. Back to the Taiga example, as shown in Figure 3.13, the priority of interoperability assessment, can be represented as the IA weights to the "cyclic dependency" enclosed by the red arrows in Figure 3.11. For instance, the higher IA weight between "Gog" and "Back" implies a higher impact on SoS integration rework effort if a suboptimal solution of integration is chosen due to insufficient interoperability assessment that was performed between them.

3.5 Evaluation and Deliverables

3.5.1 Task 3: Constructing Technical Debt Ontology

The quality of the TD ontology shall be evaluated by researchers and practitioners following an iterative process. We will publish the TD ontology and Technical Debt Ontology Sharing and Evolution (TDOSE) tool (either as an open source system or with a restrictive access within a specific user group) and allow the user community to evaluate its quality, provide feedback, and improve both the ontology and TDOSE tool through forum discussion. Before publishing the ontology and TDOSE, we will invite experts to assist us with the initial evaluation on clarity of the ontology and usability of the TDOSE tool.

Midterm Deliverable:	• The alpha version of TD ontology specification • The visualization module of TDOSE
Final Deliverable:	• The beta version of the TD ontology specification and visualization • The completed package of TDOSE with public or restrictive access

3.5.2 Task 4: Investigating Impacts of TD on SoS Integration

The evaluation of Task 4 comprises of two subtasks: (1) evaluating the impacts of TD of constituent systems on SoS integration; and (2) validating the usefulness of prioritization of SoS interoperability assessment efforts.

For the first subtask, the empirical results of how various types of TD of constituent systems would impact the SoS integration will be first evaluated internally by the TD group (both SMU and USC). Then we will evaluate the correctness and completeness of the empirical results via an online survey. In the survey, each category of TD and their potential impacts on the SoS integration will be illustrated with a few pieces of evidences extracted from the literatures. The TD categories that are commonly agreed to have impacts on SoS integration and their measurement models will be shared with USC CSSE COCOMO

III group to collaboratively define the TD adjustment factor for the next generation cost estimation model (COCOMO III).

For the second subtask, the evaluation of SoS interoperability assessment prioritization analysis will be performed on large-scale open-source software systems such as Taiga, Google. The objective of the evaluation is to verify the correctness of critical areas that are indicated by the interoperability assessment (IA) weights. We plan to conduct a set of experiments by instrumenting system interoperability mismatches to investigate how the mismatches resulted from insufficient interoperability assessment will impact the entire SoS. Finally, we will be able to compare the IA weights with the impact analysis data of the instrumented interoperability mismatches.

Midterm Deliverable:	• The impact analysis results of various types of TD in constituent systems on SoS integration • The initially defined TD adjustment factors to be integrated into the next generation cost estimation model (COCOMO III) • The evaluation results via online survey
Final Deliverable:	• The complete report on the TD impacts on SoS integration including the impact analysis of insufficient interoperability analysis

3.5.3 Task 5: Integration of TD Assessment into Next Generation Cost Estimation Models

The data collection, calibration and evaluation for the TD adjustment factor(s) will be incorporated into the calibration process of the incoming next generation cost estimation model (COCOMO III) currently with over 350 government and industrial system development data points from various domains.

Midterm Deliverable:	• The mapping of TD categories and metrics into cost drivers of cost estimation models • The upgraded cost driver rating guideline for cost estimation models
Final Deliverable:	• TD adjustment factor(s) integrated with the next generation cost estimation model (COCOMO III) that enables the estimation of extra rework effort incurred by TD remediation. • The integrative calibration and evaluation results with COCOMO III • A conference or journal paper

3.6 Project Management and Cost

The proposed three year (2016-2018) research extended from the new project incubator will be investigated in collaboration with the Center for Systems and Software Engineering (CSSE) at the University of Southern California (USC). This has been included as a part of the SERC 5-year SoSE research plan submitted jointly by USC and SMU. The estimated cost requested for the remaining research tasks (Tasks 3-5) will be $250K for each of the first two years (2016, 2017) and $200K for the third year (2018), to be split equally between SMU and USC CSSE. This funding will provide support for the SMU and USC CSSE PIs, graduate students at each university, and travel to related conferences and other project related travels. In particular, we intend to: develop a comprehensive and evolving TD ontology (Task 3); investigate various types of TD carried over from single constituent systems and their impacts on SoS integration, maintenance, and evolution (Task 4); evaluate the impacts of potential interoperability issues across constituent systems to prioritize SoS interoperability assessment tasks (Task 4); integrate TD assessment into the next generation system and software cost estimation model (Task 5). These tasks will be carried out in parallel.

In 2016, SMU will construct the initial TD ontology to facilitate the understanding of overlaps across various TD categories and different detection/evaluation methods/tools as well as to support the selection of the “best fit” metrics for TD measurement. Besides, SMU will develop the TDOSE tool based

on the latest WebGL and HTML 5 technology to support interactive TD knowledge sharing, editing and evolution. Concurrently, SMU will lead the effort to perform the empirical study on the impacts of various TD types carried over from the single constituent software systems on SoS integration, maintenance and evolution. SMU and USC will collaborate to map the metrics of various TD categories into the cost drivers in the system and software cost estimation model and to define TD adjustment factor(s) to be incorporated into COCOMO III. The COCOMO III group (CSSE) will be collecting the data, along with data on other COCOMO parameters, and incorporating the parameter into the cost model based on the recommendations from the TD group (both SMU and USC). Then the TD group (SMU and USC) will calibrate the TD adjustment factor in COCOMO III. In 2017, SMU and USC will analyze the data collected from the empirical study and cost driver mapping to upgrade the cost driver rating guidelines of current cost estimation model by integrating the TD assessment. At the same time, USC will lead the effort to evaluate the empirical results through the survey, presentations to conferences and COCOMO forum. Additionally, SMU will develop an objective approach based on DSM and SoS interoperability assessment rule space to guide the prioritization of the TD and SoS interoperability assessment efforts under resource constraints. USC will lead the effort to evaluate the existing SoS interoperability assessment models and synthesize the interoperability model into COSYSMO. **The demonstration and report of these capabilities and empirical results in 2016 and the first half of 2017 will be the mid-term exams.** In 2018, we will complete the integration of TD assessment module into the next generation system and software cost estimation model. We will also evaluate the prediction accuracy of the calibrated TD adjustment factor in the estimation model**. The final exam at the end of 2018 will be a demonstration and report of the TD adjustment factor integrated into the system and software cost estimation model with some evaluation results.** Further, we will also coordinate our research with RT-46 (Tradespace and Affordability) led by USC.

3.7 Risks and Mitigation Plan

There are risk mitigation plans built into the project to address risks associated with time lines, achieving deliverables and data collection. **Risk 1:** Project collaboration, timelines and deliverables; **Mitigation Plan**: Both SMU and USC teams will participate in biweekly Skype/WebEx meetings. Additionally, face-to-face meetings will be scheduled every 6 months to ensure the planned project progress, work breakdown structure and deliverables are followed. **Risk 2**: Data collection; **Mitigation Plan**: As mentioned in Sections 3.3.6 and 3.5.3, we will follow the seven step COCOMO II modeling methodology to save the effort and improve the quality of data collection. We will coordinate with the USC collaborator to incorporate the TD adjustment factor data collection plan into the COCOMO III data collection effort.

3.8 Conclusions

Technical debt (TD) is a growing concern for software systems. TD concerns grow even faster for SoS development and evolution. Our proposed research aims to systemize the TD concepts and measurement by developing and evolving the TD ontology. We will investigate the impacts of various types of TD from single constituent systems on SoS integration. Further, we will contribute to the development of the next generation cost estimation model via the integration of TD assessment module to support the SoS affordability decision making activities.

References

Alexander, C. (1964). *Notes on the Synthesis of Form*, volume 5. Harvard University Press.

Boehm, B., Lane, J. A., Koolmanojwong, S., and Turner, R. (2014). *The incremental commitment spiral model: Principles and practices for successful systems and software*. Addison-Wesley Professional.

Boehm, B. and Turner, R. (2003). *Balancing agility and discipline: A guide for the perplexed*. Addison-Wesley Professional.

Boehm, B. W., Clark, Horowitz, Brown, Reifer, Chulani, Madachy, R., and Steece, B. (2000). *Software Cost Estimation with Cocomo II with Cdrom*. Prentice Hall PTR, Upper Saddle River, NJ, USA, 1st edition.

Cerpa, N. and Verner, J. M. (2009). Why did your project fail? *Commun. ACM*, 52(12):130–134.

Chen, J. C. and Huang, S. J. (2009). An empirical analysis of the impact of software development problem factors on software maintainability. *Journal of Systems and Software*, 82(6):981–992.

Codabux, Z. and Williams, B. (2013). Managing technical debt: An industrial case study. In *Proceedings of the 4th International Workshop on Managing Technical Debt*, pages 8–15. IEEE Press.

Cunningham, W. (1992). The wycash portfolio management system. In *ACM SIGPLAN OOPS Messenger*, volume 4, pages 29–30. ACM.

Department of Defense. *C4ISR Architecture Working Group Final Report - Levels of Information System Interoperability (LISI)*. Washington DC: OSD(ASD(C3I)) C4ISR AWG, 1998.

Fontana, F. A., Ferme, V., and Spinelli, S. (2012). Investigating the impact of code smells debt on quality code evaluation. In *Proceedings of the Third International Workshop on Managing Technical Debt*, pages 15–22. IEEE Press.

Gartner (Accessed on Dec 01, 2014). *Gartner Estimates Global 'IT Debt' to Be $500 Billion This Year, with Potential to Grow to $1 Trillion by 2015*. 2010, http://www.gartner.com/newsroom/id/1439513.

Gery, E., Modai, A., and Mashkif, N. (Accessed on Dec 01, 2014). *Building a Smarter Systems Engineering Environment: IBM Research and Israel Aerospace Industries (IAI)*. Innovative 2010, The rational software conference, http://open-services.net/pub/Main/PlmPresentations/Building_a_Smarter_Systems_Engineering_Environment_IBM_Research_and_IAI_-ALM-1814C.pdf.

Gilb, T. (2006). No cure no pay: How to contract for software services. In *INCOSE International Symposium*, volume 16, pages 910–925. Wiley Online Library.

Griffith, I., Reimanis, D., Izurieta, C., Codabux, Z., Deo, A., and Williams, B. (2014). The correspondence between software quality models and technical debt estimation approaches. In *Managing Technical Debt (MTD), 2014 Sixth International Workshop on*, pages 19–26. IEEE.

Guo, Y., Seaman, C., Zazworka, N., and Shull, F. (2010). Domain-specific tailoring of code smells: an empirical study. In *Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering-Volume 2*, pages 167–170. ACM.

Izurieta, C., Griffith, I., Reimanis, D., and Luhr, R. (2013). On the uncertainty of technical debt measurements. In *Information Science and Applications (ICISA), 2013 International Conference on*, pages 1–4. IEEE.

Jorgensen, M. and Molokken-Ostvold, K. (2006). How large are software cost overruns? a review of the 1994 chaos report. *Information and Software Technology*, 48(4):297–301.

Kaur, A. and Kaur, K. (2013). Statistical comparison of modelling methods for software maintainability prediction. *International Journal of Software Engineering and Knowledge Engineering*, 23(06):743–774.

Kitchenham, B. A. and Charters, S. (2007). Guidelines for performing systematic literature reviews in software engineering. Technical Report EBSE-2007-01, Keele University.

Klinger, T., Tarr, P., Wagstrom, P., and Williams, C. (2011). An enterprise perspective on technical debt. In *Proceedings of the 2nd Workshop on managing technical debt*, pages 35–38. ACM.

Ktata, O. and Lévesque, G. (2010). Designing and implementing a measurement program for scrum teams: What do agile developers really need and want? In *Proceedings of the Third C* Conference on Computer Science and Software Engineering*, pages 101–107. ACM.

Lane, J. A. and Valerdi, R. (2011). System interoperability influence on system of systems engineering effort. In *Proceedings of the Conference on Systems Engineering Research*, pages 14–16.

Leon, A. (2004). *Software configuration management handbook*. Artech House, Inc.

Ligu, E., Chatzigeorgiou, A., Chaikalis, T., and Ygeionomakis, N. (2013). Identification of refused bequest code smells. In *Software Maintenance (ICSM), 2013 29th IEEE International Conference on*, pages 392–395. IEEE.

Linberg, K. R. (1999). Software developer perceptions about software project failure: a case study. *Journal of Systems and Software*, 49(2–3):177–192.

Malhotra, R. and Chug, A. (2012). Software maintainability prediction using machine learning algorithms. *Software Engineering: An International Journal (SEIJ)*, 2(2).

Marinescu, R. (2004). Detection strategies: Metrics-based rules for detecting design flaws. In *Software Maintenance, 2004. Proceedings. 20th IEEE International Conference on*, pages 350–359. IEEE.

Marinescu, R. (2012). Assessing technical debt by identifying design flaws in software systems. *IBM Journal of Research and Development*, 56(5):9–1.

Noy, N. F., McGuinness, D. L., et al. (2001). Ontology development 101: A guide to creating your first ontology.

Nugroho, A., Visser, J., and Kuipers, T. (2011). An empirical model of technical debt and interest. In *Proceedings of the 2nd Workshop on Managing Technical Debt*, pages 1–8. ACM.

Ramasubbu, N. and Kemerer, C. F. (2013). Towards a model for optimizing technical debt in software products. In *Managing Technical Debt (MTD), 2013 4th International Workshop on*, pages 51–54. IEEE.

Schmid, K. (2013). On the limits of the technical debt metaphor. In *Fourth Workshop on Managing Technical Debt*, page 4. IEEE.

Sharma, T. (2012). Quantifying quality of software design to measure the impact of refactoring. In *Computer Software and Applications Conference Workshops (COMPSACW), 2012 IEEE 36th Annual*, pages 266–271. IEEE.

Siebra, C. S. A., Tonin, G. S., Silva, F. Q., Oliveira, R. G., Junior, A. L., Miranda, R. C., and Santos, A. L. (2012). Managing technical debt in practice: an industrial report. In *Proceedings of the ACM-IEEE international symposium on Empirical software engineering and measurement*, pages 247–250. ACM.

Standish Group (Accessed on Dec 01, 2014a). *EXTREME CHAOS*. 2014, https://courses.cs.ut.ee/MTAT.03.243/2014_spring/uploads/Main/standish.pdf.

Standish Group (Accessed on Dec 01, 2014b). *THE STANDISH GROUP REPORT*. 1995, http://www.projectsmart.co.uk/docs/chaos-report.pdf.

Steward, D. V. (1981). The design structure system: a method for managing the design of complex systems. *Engineering Management, IEEE Transactions on*, (3):71–74.

Tang, R. F. and Huang, X. Y. (2011). A software project management method based on trust and knowledge sharing. *Advanced Materials Research*, 267:160–163.

Tom, E., Aurum, A., and Vidgen, R. (2013). An exploration of technical debt. *Journal of Systems and Software*, 86(6):1498–1516.

Valerdi, R., Friedman, G., and Marticello, D. (2011). *Diseconomies of Scale in Systems Engineering*. Air University.

Van Koten, C. and Gray, A. (2006). An application of bayesian network for predicting object-oriented software maintainability. *Information and Software Technology*, 48(1):59–67.

Wong, S., Cai, Y., Kim, M., and Dalton, M. (2011). Detecting software modularity violations. In *Proceedings of the 33rd International Conference on Software Engineering*, pages 411–420. ACM.

Zazworka, N. and Ackermann, C. (2010). Codevizard: a tool to aid the analysis of software evolution. In *Proceedings of the 2010 ACM-IEEE International Symposium on Empirical Software Engineering and Measurement*, page 63. ACM.

Zazworka, N., Izurieta, C., Wong, S., Cai, Y., Seaman, C., Shull, F., et al. (2014). Comparing four approaches for technical debt identification. *Software Quality Journal*, 22(3):403–426.

Zazworka, N., Shaw, M. A., Shull, F., and Seaman, C. (2011). Investigating the impact of design debt on software quality. In *Proceedings of the 2nd Workshop on Managing Technical Debt*, pages 17–23. ACM.

Zhang, H., Babar, M. A., Bai, X., Li, J., and Huang, L. (2011). An empirical assessment of a systematic search process for systematic reviews. In *Evaluation & Assessment in Software Engineering (EASE 2011), 15th Annual Conference on*, pages 56–65. IET.

Zhang, Q., Huang, L., Jan, N., Lane, J. A., and Zhang, H. (2015a). Detecting and evaluating technical debt in software systems: A systematic literature review. *Journal of Systems and Software*. Submitted at May 31, 2015, under review.

Zhang, W., Huang, L., Ng, V., and Ge, J. (2015b). SMPLearner: learning to predict software maintainability. *Automated Software Engineering*, 22(1):111–141.

Zhou, Y. and Leung, H. (2007). Predicting object-oriented software maintainability using multivariate adaptive regression splines. *Journal of Systems and Software*, 80(8):1349–1361.

APPENDIX 3.A: A TAXONOMY OF QUANTITATIVE TD METRICS BY 1) TD CATEGORIES 2) PRINCIPAL AND INTEREST

Requirements Debt

Principal:

- $\forall G \in D,\ \delta = S_1 - S_2$

Given requirement G in domain D, two specifications S_1 and S_2 at the same time t or different time t_1 and t_2, the Principal δ is the difference between two sets of tasks

Interest:

- *The increasing rate of δ*

Build Debt

Principal:

- Over-declared dependencies
- Underutilized dependencies
- Under-declared dependencies

Dependency debt ≈ Build Debt

Modularity Analysis by *CLIO*

Architecture Debt

Principal:

- $T_n = F(Ci, Cr)$

Total Cost (T) of release n is a function of implementation and rework cost, Ci and Cr

- $\forall E_k \in R,\ Ci_{total} = \sum_k Ci(E_k)$

Total Implementation Cost (Ci_{total}) is the sum of all new architectural elements (E_k) in a release (R)

- $\forall E_k \in R,\ Cr_{total} = \sum_k Cr(E_k)$
- $\forall E_j \in R_{pre},\ Cr(E_k) = \sum_j Cr(E_j)$

Total Rework Cost Cr_{total} is the sum of all pre-existing architectural elements E_k in a release (R). Each $Cr(E_k)$ is the sum of all elements (E_j) implemented in prior release.

- $Cr(E_j) = D(E_j, E_k) \times Ci(E_j) \times Pc(n-1)$

In above equation, $D(u, v)$ is the number of dependencies from u to v, and $Pc(n-1)$ is the change propagation of release $n-1$.

- *communicatesWith (External – Strctural)* Dependency
- *limitedBy (External – Behavioural)* Dependency
- *influencedBy (Internal -- Bahavioural)* Dependency
- *refersTo (External -- Structural)* Dependency
- *relatesTo* Dependency

Five types of architectural dependencies

Design Debt

Principal:

Code Smell Debt	God Class	➢ Weighted Method Count (WMC) ➢ Tight Class Cohesion (TCC) ➢ Access to Foreign Data (ATFD)
	Data Class	➢ Weighted Method Count (WMC) ➢ Number of Public Attribute (NOPA) ➢ Number of Accessor Methods (NOAM) ➢ Weight of Class (WOC)
	Duplicate Code	➢ Number of LOC that are exactly equal or just look similar

- $FIS_{flaw\_instance} = I_{flaw\_type} \times G_{flaw\_type} \times S_{flaw\_instance}$

Flaw Impact Score (FIS) is calculated by Influence (I_{flaw_type}), Granularity (G_{flaw_type}), and Severity ($S_{flaw\_instance}$)

- $DSI = \frac{\sum_{all\_flaw\_instance} FIS_{flaw\_instance}}{KLOC}$

Debt Symptoms Index (DSI) is the total FIS per Kilo Line of Code (KLOC)

- $DDSI = \sum_{instances} \left| \frac{FIS_{current\_instance}}{KLOC} - \frac{FIS_{previous\_instance}}{KLOC} \right|$

Delta DSI (DDSI) is the sum of all differences for FIS between two releases

Design Decay	Modular grime	➢ $PatC_a$: Afferent pattern coupling ➢ $PatC_e$:: Efferent pattern coupling
	Class grime	➢ No. of public methods ➢ No. of attribute
	Organizational grime	➢ C_a: Afferent coupling ➢ No. of files containing

Defect Debt

Principal:

- $\rho = \frac{p}{I_w + I_c + I_p * I_{pr} + P * I_{fr}}$

Principal (P) is the sum investigating, modification and validation cost, I_w represents interest from workaround cost, I_c represents interest from customer support cost, I_p represents interest from patch cost, I_{pr} represents probability request a patch, I_{fr} represents probability of defect eventually fixed

Automatic Static Analysis (ASA) by *Findbugs*

Code Debt

Principal:

- $TD_{principal} = (\sum V_h \times P_h \times T_h \times C) + (\sum V_m \times P_m \times T_m \times C) + (\sum V_l \times P_l \times T_l \times C)$

V_h, V_m, and V_l represent high-severity violation, medium-severity violation, and low-severity violation respectively. For each violation, Percentage (P) of work, Average Time (T) needed, and Cost (C) per time unit are necessary for calculation.

- LOC, McCabe Complexity, and Control-Flow-Nesting Level
- Number of defects during production
- Mean time taken for enhancement
- Mean time taken to fix production defects

From Developer's Viewpoint

- Count Class Coupled – No. of other classes coupled to
- Count Class Base – No. of immediate base classes
- Count Class Derived – No. immediate sub-classes
- Count Line Code – No. of lines in the class
- Count Declared Method – No. of local methods in the class

Class-level code-structure comprehension metrics

- # Sessions – No. of time-window sessions for each class
- # Class Visits – No. of time the central class is visited in a session
- # Other Class Accesses – No. unique (non-central) classes visited
- Time Spent in Class – Time spent in the central class for a session
- Time Spent in Other Classes – Time spent in all other classes

Class-level developer-behavior comprehension metrics

Self-admitted TD Comment (such as TODO, FIXME)

SQALE

- $Debt = duplication + voilations + comments + coverage + complexity + design$
- $duplication = cost\_to\_fix\_one\_block \times duplicated\_blocks$
- $violations = cost\_to\_fix\_one\_violation \times mandatory\_violations$
- $comments = cost\_to\_comment\_one\_API \times public\_undocumented\_API$
- $coverage = cost\_to\_cover\_one\_of\_complexity \times uncovered\_complexity\_by\_tests$
- $design = cost\_to\_cut\_an\_edge\_between\_two\_files \times package\_edges\_weight$

Maintenance Debt

Principal:

- Principal (P) is the estimated effort (cost) to pay the debt when it was created
- $RV = SS \times TF$

Rebuilt Value (RV) is an estimated effort for rebuilding task, decided by System Size (SS, with unit of LOC) and Technology Factor (TF, productivity factor).

- $RE = RF \times RV \times RA$

Repair Effort (RE) is calculated by Rework Fraction (RF), Rebuilt Value (RV), and Refactoring Adjustment (RA). RF is an estimate of the percentage of LOC that need to be changed; RA is decided with the experience of refactoring team, and can be removed from this equation.

- Number of bugs
- Frequency of debt-proneness
  - Number of tags bug
  - Number of reopened bugs
  - Number of duplicate bugs
- Time cost of fixing bugs

Historical data for measuring Principle

Interest:

- $IA_t = X - P$, where X is the actual effort (cost) paid for the debt.
- $IP = PC \times PM$

Interest Probability (IP) is affected by the probability (PC) of coupling of communication layers and persistence layers, and the probability (PM) that the payoff decision has to be made.

- $QF = 2^{((QualityLevel-3)/2)}$

Quality Factor (QF) is a factor used to account for the level of quality (from 1-start to 5-star).

- $ME = \frac{MF \times RV}{QF}$

Maintenance Effort (ME) is determined by Quality Factor (QF), Maintenance Fraction (MF), and Rebuilt Value (RV). MF is the amount of maintenance effort spent on systems in a yearly basis. To reach specific quality level, the extra efforts spent of this improvement is the difference of two ME.

- $RV_t = SS \times (1+r)^t \times TF$

Since the growth of TD can be represented as growth of system size, the RV will grow along with time t.

*Please find detail in "Zhang et al., 2015aZhang, Q., Huang, L., Jan, N., Lane, J. A., and Zhang, H. (2015a). Detecting and evaluating technical debt in software systems: A systematic literature review. Journal of Systems and Software. Submitted at May 31, 2015, under review."

4 Formal Methods in Resilient Systems Design Using a Flexible Contract Approach – Azad Madni, USC

Abstract

Protecting engineered systems from failure and performance degradation caused by disruptive events has been a system engineering design goal and a DOD priority for quite some time. Yet progress on this front has been slow. As a result, traditional methods of fault-avoidance and fault-tolerance are often used in conjunction with ad hoc, piecemeal resilience mechanisms. However, fault-tolerance methods typically focus on system components and the system itself. They do not address disruptions emanating from erroneous human interactions with the system, system misuse, and unanticipated system interactions with the operational environment. Furthermore, they are difficult to verify, and do not generalize or scale. This recognition provides the impetus for developing formal methods to engineer resilient systems. Formal methods enable verification, generalization and scalability. This capability is needed both by the DOD (e.g., control of distributed heterogeneous UAV swarms), and commercial sectors such as automotive (e.g., distributed, networked autonomous vehicles), energy (e.g., adaptive grids), and healthcare (e.g., patient surge handling).

Formal methods are conspicuously absent in resilience engineering for two main reasons. First, lack of research in this area has resulted in people relying on ad hoc methods such as safety nets that came out of the world of fault tolerance. Second, the definition of the term “resilience” itself tends to be wide ranging and largely context-dependent. Therefore, it is not uncommon to find the word “resilience” being used in a way that is synonymous with fault-tolerance or robust design (robust control theory). To prevent such mischaracterization, we define resilience as the ability to: recover fully/partially from the negative effects of a disruption within a specified time; dynamically extend capacity and resources to counter a disruption of a certain scale; and restructure or reconfigure the system to minimize the impact of disruptions.

The research on this incubator project is concerned with the development of formal methods to engineer resilient systems using the definitions of resilience provided above. Our approach employs a combination of flexible Contract-Based Design (CBD) and Partially Observable Markov Decision Processes (POMDP) to formally characterize complex systems and systematize the development of resilient behaviors. Intelligent control of a heterogeneous UAV swarm, a problem of significant interest within both the DOD and the commercial sector, is used to illustrate the feasibility, generalizability, and scalability of the approach. The research is intended to lead to a technology demonstration followed by one or more high payoff transitions to defense, aerospace, and space applications. The transition to defense Programs of Record (PoRs) will be through SERC and our own contacts within DOD. The transition to aerospace will be through The Boeing Company and Northrop Grumman Corporation. The transition to space will be through Jet Propulsion Lab and Aerospace Corporation.

Our team has a proven ability to engage DOD-community organizations to ensure their participation in the definition and piloting of the prototype developed on this effort. These include ERDC, ARL, and TARDEC from the U.S. Army and corresponding entities from the USAF and USN, as well as U.S. Marine Corps. It is with this transition in mind that we chose to focus on intelligent control of heterogeneous, distributed UAV swarms. We intend to coordinate with portions of RT-137 (UAV tradespace analysis, set based design). We already have written expressions of interest (see appendix A) from JPL, Aerospace Corporation, Northrop Grumman Corporation, and The Boeing Company to serve as transition partners. This report presents the accomplishments of the seedling phase of the RT-128 incubator project.

KEYWORDS: resilient systems, formal methods, contract-based design, Markov Decision Process

4.1 INTRODUCTION

Protecting engineered systems from failure or performance degradation caused by disruptive events has been a system engineering design goal and a DOD priority for quite some time (Neches and Madni, 2012; Madni and Boehm, 2014; Goerger et al, 2014)). Current approaches to resilient systems design rely on ad hoc methods (e.g., safety nets) and piecemeal solutions when it comes to developing mechanisms to respond to external disruptions and unanticipated system behaviors (Sievers and Madni 2015; Madni and Sievers 2015). In these approaches, observed high level behaviors are compared to expected high level behaviors. When the difference exceeds an acceptance threshold, the observed behavior is deemed to pose a problem, or a precursor to a problem. Such behaviors trigger a transition to a known safe state until the underlying problem is diagnosed and resolved. During that period, the system remains unusable. Furthermore, existing methods do not take into account the different states and modes of complex systems, nor do they address unprecedented disruptions that can occur at arbitrary times during system operation. They also do not address the time-dependent nature of disruptions and their impact on systems. In light of the foregoing, there is a pressing need for a formal approach to the design of resilient systems. However, the formal approach needs to have sufficient flexibility in the formalisms employed to accommodate the effects of uncertainty in system states resulting from partially observable system behavior. Thus, to advance beyond the state-of-the-art, requires the ability to determine the appropriate desired behavior of complex systems. This is difficult to do because complex systems tend to have a large state space, with some "hidden" states because of complex dynamics arising from intra-system interactions between system elements, and interactions between the system and the environment. Exacerbating the problem is the fact that the state of the system is often not known because of partial observability of the system and environmental uncertainties. Thus, while it may be possible to make predictions about physical disruptions (i.e., faults), such predictions are likely to be inaccurate. Additional complicating factors include incomplete understanding of system dependencies and environmental influences, likelihood of conflicts between local and global responses, and multiple human roles.

Formal methods offer a rigorous approach to specifying system requirements. In particular, *assume-guarantee* contracts, also called Contract-Based Design (CBD), that specify guarantees on system behaviors under specific assumptions on the environment, are especially relevant. CBD is a compositional approach to design that reduces complexity in design, implementation, and verification by decomposing system-level operations (i.e., tasks) into manageable sub-problems. In complex systems, specifications can be inaccurate or have inaccuracies, and there can be unexpected environmental factors that can complicate matters. Consequently, adaptability is crucial to ensure that the system continues to operate as intended in the face of disruptions. Formalizing the properties and behaviors of adaptive systems can help in their design, implementation and verification. Since contracts offer a general formal framework for system specification, they provide a sound starting point to build flexible contracts that are needed for resilient systems. This recognition provides the impetus for our research thrust.

The essential characteristics of a systems model, that would inform the development of resilience mechanisms, include: explicit consideration of uncertainties and risks; nominal and predictable off-nominal behaviors; unexpected behaviors (acceptable, unacceptable); different types of disruptions (internal, external); disruption attributes (duration, severity, patterns); adaptive characteristics (capacity, structure, behavior); varying levels of system observability (hidden states); and the need to perform context-driven tradeoffs (model-based, evidence-driven). What is needed today is a formal approach that enables the development and evaluation of mechanisms that protect systems against unpredictable, external disruptions. To this end, this research is concerned with developing theory-

driven formal approaches, and model-based methods for engineering resilient systems (Madni and Sievers, 2015b).

The remainder of this report presents the overall approach, the underlying formal methods, the key characteristics of the methods, and their application to multi-UAV swarm control, an application of interest to both the DOD and civilian sectors. The report concludes with a plan for project execution and transition in a potential follow-on.

4.2 PROMISE OF FORMAL METHODS

Formal methods consist of techniques that are used to model complex systems in a mathematically rigorous fashion. The resultant models enable verification of the system's properties more thoroughly than would be possible using purely empirical testing. With formal methods, there is an important tradeoff between the level of rigor and the degree of flexibility in the model to capture complex behaviors. For example, both system safety and resilience become important considerations. In this regard, a formal approach can effectively complement system testing to ensure correct behavior (Madni and Sievers, 2015b). In contrast to traditional system design methods, formal methods employ formal verification schemes to ascertain that the basic principles governing system behavior are proven correct before they are accepted. It is important to note that formal verification cannot and does not circumvent the need for testing because formal verification cannot fix unwarranted (i.e., poor) assumptions in design. However, formal verification can help identify reasoning errors that would otherwise go unresolved (i.e., left unverified).

Formal methods can be deterministic or stochastic. The choice of modeling approach depends on factors such as system observability, system controllability, need for adaptability, and uncertainties arising from environmental unpredictability. It is important to realize that the choice of deterministic or stochastic methods is not an either-or proposition. The two approaches can co-exist and, in fact, can be combined to exploit the benefits of each while circumventing their respective limitations. Examples of deterministic methods include computational tree logic, linear temporal logic, and contract-based design. Examples of stochastic methods are Partially Observable Markov Decision Processes (POMDP) and Hidden Markov Models (HMM). These are black box models in the sense that the underlying states, state transitions, and state emissions are unknown or partially known at the outset. However, these models can be trained to recognize nominal behavior and flag unusual behavior through the use of appropriate learning techniques.

4.3 APPROACH

Our approach is rooted in three key advances: a) different views of engineered resilience; b) resilience contracts (RCs) that extend CBD to accommodate flexible assertions; and c) integration of RCs with POMDP modeling formalism for system state detection and correction. These key elements of our approach are discussed next.

4.3.1 ENGINEERED RESILIENCE

Engineered resilience is a system property that allows a system to continue to provide useful service in the face of largely unpredictable, disruptive events which can be internal or external to the system. The types of disruptions fall into three categories: external disruption – caused by factors outside the control of the system; systemic disruption – service interruption due to an internal fault; and human-triggered disruption – the result of human error or system misuse (Madni and Jackson, 2009). System behaviors that are currently viewed as resilient behaviors are:

- *Circumvent disruption* – this means the system has the ability to anticipate and avoid having to confront the disruptive event; this capability has been addressed in the body of work called fault avoidance and obstacle avoidance.
- *Withstand disruption* – this means being able to confront and endure the disruption without degradation in performance within the system's performance envelope; this capability has been addressed in the body of work called robust control.
- *Recover from negative effects of a disruption* – this means the system suffers temporary degradation in performance but is able to recover from the disrupting event to an acceptable degree and within an acceptable duration; that both the disruptive event and system response can potentially leave both the system and environment changed from their pre-disruption states.
- *Dynamically extend capacity* – this means adding resources on-demand to counter the disruption; the added resources may be released post-disruption, or repurposed.
- *Restructure/Reconfigure system* – this means altering the participating components (or nodes in a SoS), as well as linkages and information flows among them to minimize/mitigate the impact of the disruptive event to assure continuity of system operation/service (e.g., cybersecurity breach).

Of course, system failure implies that the system is incapable of providing useful service. Of the five types of resilient behaviors presented above, the first two are covered by existing methods from fault avoidance and robust control theory. The remaining three require advances in theory, concepts and new methodologies. From our perspective, these three collectively fall under the rubric of engineered resilience. The last three also require real-time tradeoffs. For example, to recover from negative effects of a disruption, there is a tradeoff between the degree of recovery and the time to recovery. Similarly, to dynamically extend capacity there is a tradeoff between the amount of added capacity and the cost of adding that capacity. Finally, in restructuring or reconfiguring the system, there are tradeoffs involving time to restructure/reconfigure, degree of restructuring/reconfiguring, the added complexity from the restructuring/reconfiguration and the cost of restructuring/reconfiguration. It is the latter three resilient behaviors that we are targeting in the development of formal methods.

4.3.2 BUILDING BLOCKS AND EXTENSIONS

Contract Based Design (CBD) is a formal method for specifying system requirements, behaviors, and implementations. A contract comprises a pair of invariant assertions, a statement of an input condition, and a guaranteed system behavior under those conditions. The approach is compatible with proofs of correctness, decomposability, tradespace analysis, and online error monitoring. For these reasons, CBD is well-suited for describing large-scale systems and for building fault-tolerant mechanisms. Table 4.1 presents a description of contract based design using mathematical notation.

Table 4.1. Contract Based Design

• A contract, C, is defined by a pair of assertions, C = (A, G), in which A is an assumption made on the environment and G is the guarantees a system makes if the assumption is met.
• For example, a system is guaranteed to produce an output from the set $o \in \{ o_o, o_1, \ldots o_{n-1}\} \subseteq O$ when in the state $\sigma \in \{\sigma_o, \sigma_1, \ldots \sigma_{n-1}\} \subseteq \Sigma$ for an input $i \in \{i, i_1, \ldots i_{m-1}\} \subseteq I$ where O is the set of all outputs, Σ is the set of all system states, and I is the set of all inputs

(Sangiovanni-Vincentelli et al., 2012), (Meyer, 2000), (Le Traon et al., 2006)

These are formal, checkable system representations that have been successfully used for defining and validating error detection mechanisms.

Partially Observable Markov Decision Processes (POMDP) is a special case of the Markov Decision Process. It is well-suited to describing many real world problems and situations that are neither fully observable nor controllable. Interestingly, the Markov assumption is often valid for real world problems and systems. A POMDP is defined by: a set of states; a set of actions; a set of observations; and a transition model, reward model, and observation model. The Markov assumption associated with the transition model implies that the optimal policy depends only on the current state. The fact that the operational environment is only partially observable means that the current state is not necessarily known and, therefore, an autonomous agent cannot execute the optimal policy for that state. The general idea is that based on the belief state, the agent can act, judge the impact of that action, and adjust the response.

Incorporating Flexibility in POMDP: Flexibility can be incorporated in POMDP by (a) relaxing the time invariance restriction on the state space and/or action space; (b) adding an evaluation metric to determine best action; (c) updating emission and transition properties of hidden states; and (d) adding the concept of time. By relaxing time invariance restriction on the state space and action space allows, the model to adapt to actual system behavior. Relaxing this restriction, also allows state transition probabilities to be adaptable, while allowing for emergent states, and accounting for unobservable and uncontrollable states. The addition of an evaluation metric allows the determination of the best action when the system is believed to be in a particular state under a certain set of assumptions. Updating emissions and transition probabilities determine which outputs are observable and what actions are performed. After sufficient model "training," high confidence can be developed in resulting model probabilities. The model emits observable outputs and internally used action controls. Finally, to add the concept of time, we specify the number of observations before a transition can occur can be specified, and the evaluation and parameter estimation functions used in model training can be appropriately modified.

4.3.3 SOLUTION APPROACH

Our solution approach is based on a hybrid representation that employs: a formal deterministic representation to assure rigor; and self-adapting mechanisms to provide the requisite flexibility in continuously monitoring system behavior and incrementally learning initially unobservable states. The specific modeling methods that we employ are a Resilience Contract (RC), a flexible variant of CBD and POMDP to serve as the self-adapting mechanism. The POMDP is well-suited to modeling "hidden states" and can be trained to reflect system behavior. The POMDP model is evaluated against system behavior, and the probability that the model output matches system behavior can be computed along with the identification of the most probable system state. If the model and system outputs agree, then the system state can be inferred, and a path back to a normal or safe state can be pursued (recovery). If there is disagreement between the trained model output and system behavior, then it can be concluded that a previously unrecognized condition has occurred, and either retrain the existing model (i.e., change transition and emission probabilities), or add a new state if observed behavior is "distant" using Mahalanobis distance from existing system states.

For a traditional contract, an implementation is said to satisfy a design contract if it fulfills guarantees when the assumptions are true. While subsuming fault-tolerance, a RC employs flexible assertions that are adaptable and that can also be probabilistic. This feature allows a RC to respond dynamically to

unexpected disruptions. A RC adds flexibility to deterministic contracts to generate adaptive system response to a disruption. This characteristic is at the heart of resilience mechanisms. A RC exploits in-use learning, and is capable of uncertainty handling and pattern recognition. It is also important to recognize that formal deterministic models are not adaptive in the sense that they are defined by a set of invariant assertions and pre-conditions that fully define the domain of legal inputs and post-conditions. The post-conditions are either correct for legal inputs, or an error is declared. However, for self-adaptive models, we define a new concept that allows for incomplete specification of legal inputs and a flexible definition of post-condition correctness. These characteristics are the essence of a resilience contract (RC).

Fundamental Concepts. Classical fault-tolerance is represented by "inflexible" assertion. That is, in classical fault-tolerance, we make assertions in which assumptions are based on a priori analysis of disruptive events. The contract then guarantees what we believe best corrects the problem situation, or leaves the system in a safe configuration. The primary issue, however, is that in a complex system, system state is seldom known. Therefore, we cannot know with certainty whether an action we take will improve or degrade system operation. Consequently, our approach addresses uncertainty by evaluating the impact of small decisions that move a system from an unknown/problematic state to a healthy or safe state.

Resilience Contract in Operation. Let us assume that the complex system appears to be operating reasonably for the most part but there are indications that not everything is fine. The questions that arise are: Is the system operating ok? Are there signs that the system could be headed for trouble, if certain actions are not taken? Could the system be already in trouble, but we are unaware of that? To answer such questions, we need flexibility in assertions (i.e., have flexible assumptions). Rather than have fixed assumptions about faults, typical of fault-tolerance, we employ a Markov Decision Process (MDP) to evaluate belief about system state. Belief states are initialized with design values and updated during online learning. A policy (which is learned and updated during system operation) based on belief states directs responses which might include: continue to monitor (collect more data), take an action (assert a response and observe what happens), and safe the system and solicit help.

Making the Right Choice. If the choice is right, i.e., if the policy directs the correct action, then there will be a noticeable or no discernible improvement in system condition. The latter could imply that the system is stable despite some indications to the contrary, and so continued monitoring is appropriate. Also, correctness of choice can be reinforced by increasing confidence in the choice. This is accomplished by changing transition parameters probabilities in the belief model and/or reward parameters for actions.

Making the Wrong Choice. If the choice is wrong, then either the belief MDP is wrong, or the policy is wrong. In either case, a new belief state can be computed, and the action directed by the policy at that state taken. The belief MDP is updated as needed to reflect new knowledge regarding the system state, and the policy is changed accordingly. In the event that a number of attempted actions do not work, a general safety-net response can be pursued.

Controlling POMDP and Dealing with Non-Determinism. When a probabilistic system interacts with the environment, non-deterministic choices are possible in addition to probabilistic moves. Such choices are captured by MDP, which extend Markov Chains with non-determinism. However, in several problem scenarios, the system is not observable and the information about system state at a given instant is not previously known. The presence of such uncertainty in observations can be captured by POMDP. The control of an MDP means defining a policy, i.e., a function that associates with system history a distribution on non-deterministic choices. The steady state control problem for MDP is a well-studied problem that is decidable, and can be represented as a linear program that is solvable in polynomial

time. However, with partial observability in POMDP the steady state problem becomes undecidable. Thus, a way to deal with non-determinism is needed.

The POMDP construct enables evaluation of most likely system state at any point in time. It is also possible to determine a measure of how confident we are that the most likely state is correct. In this regard, a state transition probability graph developed during nominal and off-nominal operations guides an action decision process. The resultant graph includes one or more "safe states" defined by invariant assertions that serve as a goal for off-nominal conditions, i.e., those either known as off-nominal, or previously unobserved. In off-nominal conditions, actions are trajectories toward nominal states when possible, or safe states when not. The impact of actions taken may result in a new most likely state which, in turn, may result in new actions, or continuation of current actions.

Verification. Because the approach relies on in-use training, it is not possible to verify that the model covers all possible disruptive events. However, rigorous modeling of contract assertions will enable to perform consistency and reachability checks. We intend to demonstrate this capability in the follow-on phase. In particular, a key challenge in hierarchical resilient systems is assuring that resources needed for specific actions are available when needed, and that the different levels within the hierarchy cooperate. While we have not fully explored the option space that accomplishes the checking, resource availability, and conflict avoidance goals, in many ways the problem is similar to that tackled by linear temporal logic (LTL) methods that rely on Büchi automata (BA) for checking. A concept currently under consideration modifies the BA to include hidden states and unknown transition probabilities. A common checking paradigm checks that the intersection of the language produced by a BA that represents the system, and the language provided by a BA representing the complement of system assertions, is null. Our follow-on research will evaluate the effectiveness of the above approach under the condition of probabilistic and hidden assertions and explore other methods, as necessary.

4.4 SEEDLING STUDY: HETEROGENEOUS UAV SWARM CONTROL

The goal of the seedling study was to develop an adaptive resilience strategy for a swarm of small UAVs that collectively fly to a location, collaboratively perform the mission, and return. These capabilities are distributed and shared among the members of the heterogeneous swarm, rather than co-located within a single, multi-sensor, multi-function UAV. The impact of this architecture is lower cost, smaller UAV size, and on-demand adaptability to respond to disruptive events. The latter is the key to a resilient swarm, in which each UAV self-adapts to unexpected conditions en route and during the conduct of the mission. Our research specifically addresses fine-grain swarm planning and adaptation to accommodate disruptive events. In our system concept, higher level plans and goals are determined by an external entity (e.g. mission headquarters), while the swarm self-determines localized adaptations that have the highest likelihood for achieving higher level goals in response to perceived threats and other disruptive events. The overall concept is similar to that of driving a rover on Mars. In this concept, the ground station determines high level goals, the rover samples the environment, and based on those observations determines how best to achieve those goals. The primary difference from the Mars rover concept is that the swarm self-reconfigures resources based on mission goals when failure occurs. The key innovation in this approach is the design of self-adaptive, *flexible design contracts* that include *stochastic assertions* and *actions*.

UAV Swarms are an area of study with interest from both the DOD and the civilian sector (e.g., agriculture, search and rescue, locating and tracking chemical clouds) as well as the field of complex adaptive systems. Therefore, this domain has significant payoff as a SERC research domain. Scheutz et al (2005) investigated a UAV model for locating and tracking chemical clouds. Their concept exploits biologically-inspired agents that exhibit and accommodate emergent behaviors as the swarm maneuvers

towards its goal while individual UAVs collaborate to avoid collision. The agent model comprises six rules expressed as "contracts" (Table 4.2).

Table 4.2. Agent Model Rules as "Contracts"

• If chemical, then activate attraction beacon • If no chemical, then deactivate attraction beacon • If UAV within collision range, then turn away • If no UAV within collision range, turn right (left) • If no UAV within collision range, update turn decision

Another relevant publication is on multiagent swarming system for distributed Automatic Target Recognition (ATR) using UAVs (Dasgupta, 2008). This researcher describes an ant colony algorithm that uses a "pheromone" trail for directing UAVs to a high value location (Figure 4.1). In an ant colony, there is chemical signaling to relay particular conditions to the rest of the group. Using this analogy, we use messages to inform the rest of the group. We employ the "cooperating specialists" paradigm in which each UAV is a specialist that cooperates with the other specialists to work towards mission objectives. In the ant colony example, there are ants that defend, there are ants that forage, and there are ants that take care of the queen. In our UAV swarm example, there are UAVs that look for potential conflicts, there are UAVs that ensure getting to the destination in the most cost-effective way, and there are UAVs that perform housekee ound station.

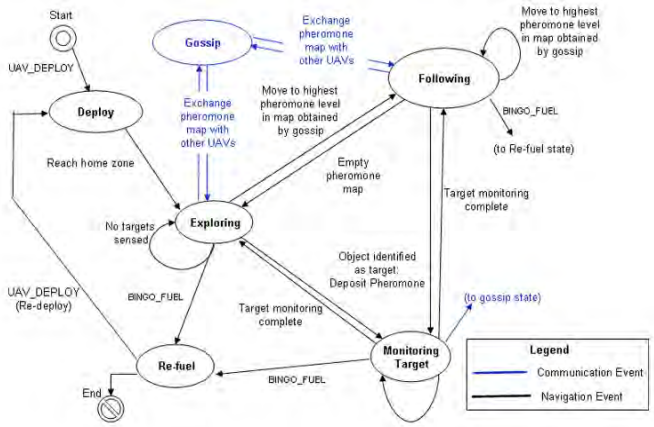


Figure 4.1. Multiagent Swarming System for Distributed ATR

Wei et al (2013) developed a simulation framework for UAV swarm configuration and mission planning. This simulation framework accommodates mission goals, individual positions, and adaptable trajectories. Recently, the U.S. Navy demonstrated an autonomous UAV swarm concept under ONR-sponsored research. http://defensesystems.com/articles/2015/04/15/onr-locust-swarming-autonomous-uavs.aspx. An example of a heterogeneous UAV swarm is shown in Figure 4.2. The swarm comprises Command and Data Handling (CDH) UAVs and Mission UAVs that communicate though a wide-band (WB) uplink and Narrow Band (NB) Tracking, Telemetry and Command (TTRC). The CDH UAV is capable of: communication; real-time plan changes; plan execution and adaptation; swarm formation, control and status, and coordinated emergency response. The distributed functionality for this UAV type encompasses space and ground communication radios, onboard plan, mission data collection, mission data transmission, swarm communication, swarm control, swarm Health Status and Accountability (HSA), swarm resilience, swarm alarm and status, Attitude Determination and Control System (ADCS), and environment analysis and response. The mission UAVs are capable of imaging and image processing,

threat evaluation, threat response, targeting, and damage assessment. Their distributed functionality encompasses ground optical imaging, ground radar, ground IR, sensor fusion and processing, weather sensors, threat sensors, air-to-air defense, air-to-ground defense, CDH drone communication and ADCS.

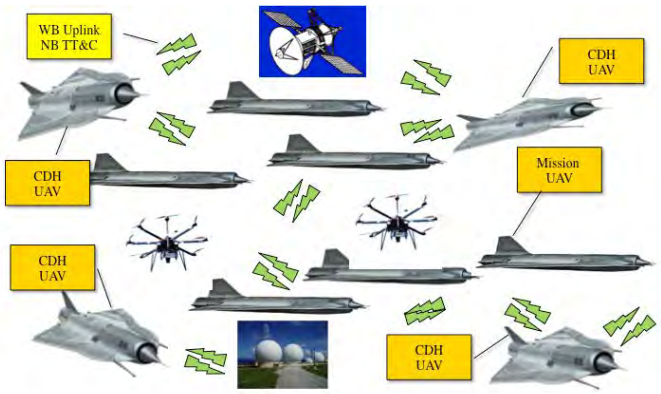


Figure 4.2. Example of a Heterogeneous UAV Swarm

The goal of the heterogeneous swarm is to fly to the target using a pre-loaded flight-path, and acquire and transmit sensor data for use by command and control (C2). The mission plan is defined by a set of flexible contracts that we call resilience contracts (RCs). RCs enable the swarm to evaluate its current state and determine which options have the best outcome, given current state and likelihood of success. We use POMDP as the basis for RC. As noted earlier, the POMDP is a decision process in which system dynamics are assumed to follow a Markov Decision Process (MDP) with some hidden states. It is a memoryless decision process based on evaluating transition rewards. In POMDP, some of the states are not observable (i.e., hidden) because of uncertainties regarding the outcome of an action, and uncertainties regarding the environment because of imperfect information. In CBD (Table 1), A and C are invariants in which assumptions are preconditions and guarantees are post-conditions. A RC extends invariant contracts in three ways: (a) models unobservable states; (b) looks for emergent behavior; (c) employs the triggered action itself as a contract.

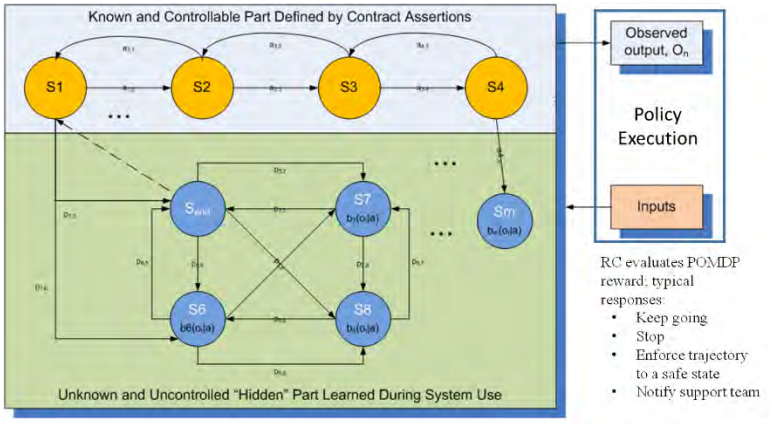


Figure 4.3. Resilience Contract Agent

Flexibility is achieved through the introduction of POMDP representation within the RC agent (Figure 4.3). As shown in this figure, a RC agent has deterministic and stochastic parts. The agent accepts inputs and generates observations. The agent generates observable outputs while operating with partially observable states. The inputs to and outputs from the RC agent defines the execution policy space. The

RC agent evaluates the POMDP reward and responds accordingly. Typical responses are: keep going; stop; enforce trajectory to a safe state; and notify support team.

The POMDP maintains a probability distribution over all states transitions and observations. The probability distribution is used to evaluate what is the most likely state of the system and the reward(s) for making such decisions. Typically, an "agent" performs this evaluation and makes decisions based on a "policy." The policy determines which actions to take in a given belief state. The agent in this case is the RC while the policies are the basis for decisions. Since it is not practical to define all policies in a complex system, RCs add the management of previously unknown conditions to traditional contracts. The first step in modeling is creating a POMDP model. An exemplar POMDP is shown in Figure 4.4.

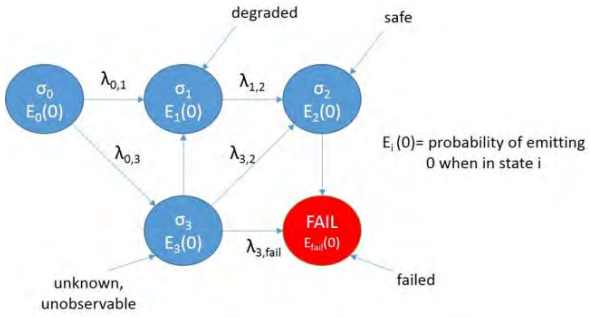


Figure 4.4. Exemplar POMDP

This model, which includes observable and unobservable states, is trained during system use by determining transition and emission probabilities. During use, system outputs are used to estimate the state that maximizes the posterior probability from the observations as well as the maximum likelihood that the POMDP parameters could have produced the observations. For states known to be problematic or potentially problematic, the POMDP evaluates a reward function that determines whether an immediate action or delayed action is expected to produce the best outcome. Actions are intended to move the system away from problematic states to desirable or safe states. For example, transition $\lambda_{3,1}$ is an example of a transition from a "bad" state to a good state. The next question is how to accommodate emergent behaviors, i.e. those that were not previously observed. Previously unobserved disruptions or partially observable disruptions can lead to previously unknown states. These states can give rise to emergent behavior. Emergent behavior, due to previously unknown states, is evaluated by computing the statistical distance to prior known states. When the weighted distance (i.e., Mahalanobis distance) exceeds a limit, a new state is added. This algorithm (Lee et al, 2010) avoids conditions in which the POMDP predicts a strong match to a prior known state even though the observation mean is far from that state. Figure 4.5 presents an example of the swarm control architecture.

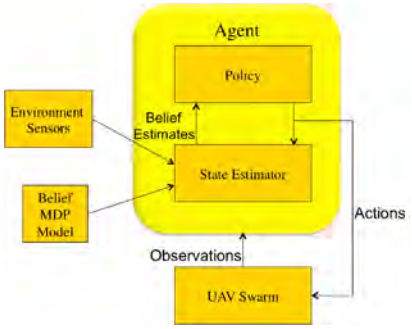
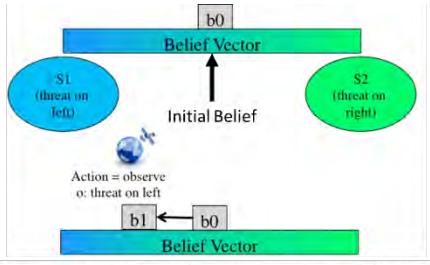


Figure 4.5. Example Swarm Control Architecture

As shown in this figure, swarm control is based on creating an optimal policy based on belief estimates provided by the state estimator. The state estimator relies on observations from the UAV swarm, environment sensors, and MDP belief model to generate updated belief estimates. Policy actions act on the UAV swarm and are used by the state estimator to update state information.

A simple CONOPS for a UAV swarm is used to illustrate the approach. In this scenario, the UAV swarm needs to turn left or right to avoid an obstacle. There is uncertainty regarding the location of the threat. The threat could be to the left or the right of the swarm. A decision needs to be made to veer left or veer right. If the swarm veers right and the threat is located/headed to the right, serious consequences could ensue. The same is true if the swarm veers left and the threat is located or headed to the left. There are three possible actions that the swarm can take: veer left; veer right; continue flying straight ahead while continuing to collect more data on the threat. The POMDP policy for this simple CONOPS has to deal with a variety of considerations such as: UAVs not inadvertently crash into each other; all UAVs get safely to their destination; UAVs avoid potentially disruptive events; if one or more UAVs is shot down, the remaining UAVs need to reorganize and reallocate functionality to ensure achievement of objective to the extent feasible. The key ideas behind an optimal POMDP policy are two-fold: a POMDP policy maps current belief into an action; and an optimal POMDP policy is a continuous solution of a belief MDP. Figure 4.6 shows the equation for summation of outcomes based on the path the UAVs take. The equation normalizes the rewards and penalties. As shown in Figure 4.6, the system starts with a 50-50 belief that the threat could be to the left or the right. The system then makes an observation of a potential threat to the left. The system revises its belief from b0 to b1, i.e., there is a greater belief that the threat could be to the left. The belief is updated in accord with Bayesian analysis using observation and current state.



$$b\downarrow 1\ (s\downarrow i)=P(o|s\downarrow i,a)\sum s\downarrow j\in S\uparrow P(s\downarrow i|s\downarrow j,a)b\downarrow 0\ (s\downarrow j)\ /P(o|a,b))$$

Figure 4.6. Iterative Update of Beliefs

A key problem with such state space models is that they are subject to combinatorial explosion. To contain this explosion, several methods can be potentially applied including: Pruning (Bellman equation), branch and bound, heuristic search, Monte Carlo search, and policy tree.

4.5 Project Work Plan and Potential Transition Partners

In the follow-on phase, we plan to further formalize the mathematical models outlined in the seedling phase. To this end, **we will coordinate our research with portions of RT-137 (AFIT and NPS research for ISR UAV tradespace analysis, and possibly with Wayne State and Penn State research on Set Base Design).** In particular, we intend to: develop a rigorous contract model-based on the POMDP construct; evaluate the impact of flexibility on formal checking methods; and develop practical constraints and methods for applying RCs using a heterogeneous UAV swarm as an example. In 2016, we will extend the POMDP representation with confidence levels to state estimates, to impact belief maximization and subsequent actions. We will also introduce means to incorporate state additions as actions in the

POMDP (ROM cost: $225K). In 2017, we will incorporate means to deal with multi-state, multi-observation POMDP-based RCs. **The demonstration of these capabilities in 2016 and 2017 will be the mid-term exams**. We will demonstrate this capability to the sponsor and transition partners (ROM cost: $225K). In 2018, we will incorporate the means to integrate local UAV actions into swarm actions. We will also incorporate means for modeling UAV swarm that incorporates RCs. We will demonstrate this capability to transition partners (ROM Cost: $180K). In 2019, we will pursue these transitions. **The final exams at the end of 2018 and 2019 will be a showing of integration of local actions into swarm actions, and one or more successful transitions, respectively**. To this end, we will generate applicable domain model(s) for customer domains (ROM Cost: $150K).

REFERENCES

Dasgupta, P., "A Multiagent Swarming System for Distributed Automatic Target Recognition Using Unmanned Aerial Vehicles," IEEE Trans on Sys, Man, and Cybernetics, Part A – Systems and Humans, 38 (3) May 2008

Goerger, S.R., Madni, A.M., and Eslinger, O.J. "Engineered Resilient Systems: A DoD Perspective," *Conference on Systems Engineering Research (CSER 2014)*, Eds.: Azad M. Madni and Barry Boehm, Procedia Computer Science, Elsevier, 2014.

Le Traon, Y., Baudry, B., "Design by Contract to Improve Software Vigilance," IEEE Trans. on Software Engineering, Vol. 32, No. 8, Aug. 2006

Lee, et. al., "Online Degradation Assessment and Adaptive Fault Detection using Modified Hidden Markov Model," Journal of Manufacturing Science and Engineering, APRIL 2010, Vol. 132

Madni, A.M. Expanding Stakeholder Participation in Upfront System Engineering Through Storytelling in Virtual Worlds; *Systems Engineering*, Vol. 18, No. 1, pp. 16-27, January 2015.

Madni, A.M., Sievers M.W., "A Flexible Contract-Based Design Framework for Evaluating System Resilience Approaches and Mechanisms," IIE Annual Conference and Expo, ISERC 2015, May 30- June 2, 2015

Madni, A.M. and Boehm, B. (eds), "Engineered Resilient Systems: Challenges and Opportunities in the 21st Century, Procedia Computer Science 28 (2014), ISSN 1877-0509, Elsevier, 2014.

Madni, A.M. and Jackson, S. Towards a conceptual framework for resilience engineering. *IEEE Systems Journal,* 3.2, 181-191, 2009.

Madni, A.M., and Sievers, M. Model Based Systems Engineering: Motivation, Current Status and Needed Advances, accepted for publication in *Systems Engineering*, 2015.

Meyer, B. "Towards More Expressive Contracts," J. Object Oriented Programming, pp. 39-43, 2000

Sangiovanni-Vincentelli, A., Damm, W., and Passerone, R. "Taming Dr. Frankenstein: Contract-Based Design for Cyber-Physical Systems," European Journal of Control 18 (3): 217-238, 2012

Scheutz, Schermerhorn and Bauer, "The Utility of Heterogeneous Swarms of Simple UAVs with Limited Sensory Capacity in Detection and Tracking," Swarm Intelligence Symposium, 2005. SIS 2005. 2005 Proc. of IEEE

Sievers, M.W. and Madni, A.M., "A Flexible Contracts Approach to System Resiliency," *IEEE Systems, Man and Cybernetics International Conference,* invited special session "Frontiers of Model Based Systems Engineering", San Diego, CA, Oct 5-8, 2014.

Wei, Blake, and Madey, "An Operation-time Simulation Framework for UAV Swarm Configuration and Mission Planning," Procedia Computer Science 18 (2013) 1949-1958

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109-8099
(818) 354-4321

May 14, 2015

To Whom it May Concern,

JPL is very interested in the resiliency work that Professor Azad Madni is conducting at USC, for our future space flight missions. Our future missions will explore planets, moons, comets, and other planetary objects for which very little is known about the environment where our spacecraft will operate. While surprises are exciting for scientists, they pose serious challenges to designers in the form of unpredictable disruptions, i.e., unknown-unknowns. Currently we depend on ad-hoc methodologies for addressing unknown-unknowns and cannot completely evaluate vulnerabilities. As the Lead for establishing JPL's Strategic Plan for Spacecraft Autonomy, I recognize that the work Professor Madni is pursuing in the area of formal methods for defining and evaluating resiliency could have a significant impact on the success of our future missions. JPL is exploring possible collaboration opportunities with Professor Madni in this research area through programs such as our Strategic University Partnership Program.

Sincerely,

Lorraine M. Fesq

Lorraine M. Fesq, Ph.D.
Chief Technologist, Systems Engineering and Formulation Division
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

NORTHROP GRUMMAN

Aerospace Systems
Northrop Grumman Corporation

3520East Avenue M,
M/S 4B 71HA
Palmdale, CA 93550

July 10, 2015

To Whom it May Concern,

Professor Azad M. Madni has a long relationship with Northrop Grumman Corporation. I follow his research in complex systems and engineered resilience with great interest. Without a doubt, he is one of the top researchers in the country in these areas. He has a unique vision that frequently leads to ground-breaking research. In particular, he is very well-known in Engineered Resilient Systems research in the last decade. I believe his DOD-SERC-sponsored research in Formal Methods in Resilient Systems Design Using A Flexible Contract Approach has huge potential. For NGC, this research can produce significant benefit to our current and future developments in multi-UAV operations. Unmanned Aerial Systems often times operate in uncertain environment in which they are exposed to various forms of disruptions (e.g., cyber-attacks).

Complex systems such as UAV swarms clearly need to be resilient. Piecemeal ad hoc solutions will not serve our purpose. We need well-defined formal method to address existing challenges in modeling and designing resilient systems. The work that Professor Madni is doing for DOD-SERC is highly relevant to our programs an IRAD in multi-UAV systems.

I have an interest in his research and have addressed this need with my leadership to serve as a transition partner and pilot site for this promising technology. I believe we can exploit the findings from this research in both ongoing and future programs in this area.

Michael E. Wallace

Michael E. Wallace, BSEE, MSEM, EISE
Avionics Engineering Department Manager
Northrop Grumman
m.wallace@ngc.com
661-540-0290
661-400-1181 mobile

AEROSPACE
Assuring Space Mission Success

July 09, 2015

Dear Colleagues,

Professor Azad M. Madni at the University of Southern California Systems Architecting and Engineering Program has been conducting ground-breaking research in Engineered Resilient Systems for nearly a decade. At The Aerospace Corporation, we have been following his research during this time period. His recent publications in ISERC 2015 and IEEE SMC 2014 were closely reviewed by members of our technical staff. We believe his SERC-sponsored research in Formal Methods in Resilient Systems Design Using a Flexible Contract Approach could be of significant benefit to our current and future missions in National Security Space (NSS) programs. NSS systems operate in uncertain environments in which they are exposed to various forms of disruptions. Thus, the ability for these systems to exhibit resilient behavior is a key attribute for architectures and designs. Currently, there is no well-defined formal method to address the existing challenges in modeling and designing resilient systems.

Professor Madni's research in this area is very relevant to our corporate strategic initiatives. As such, we are most interested in evaluating the technology and identifying potential opportunities to apply it. We would be interested in exploring serving as a transition partner and being a pilot site for this technology.

Sincere regards,

Marilee J. Wheaton

Marilee J. Wheaton
Fellow of AIAA and INCOSE
Systems Engineering Division
The Aerospace Corporation

July 1, 2015

Dr. Azad Madni
Professor, Astronautical Engineering
Technical Director, Systems Architecting and Engineering
Viterbi School of Engineering
University of Southern California
Los Angeles, CA, 90089

Subject: The Boeing Company's support for the RT-128 DOD SERC proposal titled: "Formal Methods in Resilient Systems Design using a Flexible Contract Approach

Dear Professor Madni,
We have read your research report entitled, "Formal Methods in Resilient Systems design using a Contract-Based Approach." We find your research approach to be exciting and potentially ground-breaking. The four key aspects of your approach that stood out for us are:

- The pursuit of rigor through the use of Contract based Design to enhance verification and scalability;
- the focus on incorporating flexibility in the contracts so that the system exhibits resilience in the face of systemic, external, and human-triggered disruptive events
- the focus on a real world application (i.e., heterogeneous UAV swarms) of great national interest; and
- The emphasis on transition and pilot evaluation of the overall concept.

As you are aware of, The Boeing Company is committed to pursuing the promising discipline of resilience engineering as a means to counter unexpected and unknown conditions and events, and as a means to conquer system complexity. Our leaders and technical experts are interested in methods that do not rely on ad hoc resilience mechanisms. Your proposed effort will help us in these initiatives especially in evaluating effective practices in production engineering and complex systems design, two of our central concerns. Your emphasis on verifiability and scalability on the one hand, and affordability and formal methods on the other, are especially appealing to us.

We would like to partner with you on this exciting incubator effort as a transition and evaluation partner, and a potential co-researcher on this effort. We wish you great success in securing follow-on funding from the DOD SERC and look forward to working with you and your team in this important area.
Sincerely,

Michael C Richey

Michael Richey, PhD
Associate Technical Fellow
The Boeing Company
Complexity, Learning Sciences and Engineering Education Research
International Cell: 425-750-4635
Email: michael.c.richey@Boeing.com

5 FOUNDATIONS OF SYSTEMS ENGINEERING – KEVIN SULLIVAN, UVA

5.1 INTRODUCTION

Systems engineering stands on shaky foundations. This weakness infuses both the theory and the practice of the discipline.

New abstract models, e.g., suggesting novel approaches to defining system properties, are often presented in an astonishingly informal manner. This has the benefit of flexibility, but the disadvantages of ambiguity, difficulty assuring logical and mathematical soundness, and the impossibility of using such models to support either precise and testable system specifications, or automated mechanisms for managing them.

In *practice*, requirements that cover the full range of critical system properties relevant to stakeholder value objectives and constraints are impossible to adequately specify and verify today. By *requirements* we mean *objectives* and *constraints* on outcomes produced by a system in dimensions of value to stakeholders as well as derived constraints on system properties required for the system to produce satisfactory outcomes.

Properties span the gamut of technical concerns, ranging from cyber and physical functionality to dependability, usability, evolvability, resiliency, affordability, adaptability, and many more. While the state of the art in the specification, realization, verification, and assurance of functional properties is well developed, if not entirely adequate, the state of the art with respect to the many other non-functional properties (also called *ilities* or *qualities* or *quality attributes*) is exceptionally weak. Yet these are precisely the properties whose characteristics and tradeoffs pose the greatest risks.

A big part of the problem is that we continue to lack taxonomies of system properties sufficient to support comprehensive specification, verification and assurance. Decades of informal attempts to define properties have left us with still informal and relative vague notions, for the most part, and lacking in notations capable of supporting rigorous definition, specification, verification, or assurance in these dimensions. Tradeoffs involving such properties are hard to anticipate, understand, recognize, and manage. Solutions are hard to design, verify, and change, often due to unreasonable vagueness in major system requirements and specifications.

5.1.1 GOALS AND APPROACH

The goal of this seedling project was to take first steps toward a project the aim of which would be to develop a strategic new approach to strengthening the foundations of systems engineering. The approach would revolve around an effort to integrate a diverse range of *formal methods* into both systems engineering practice and the development, testing and application of new system engineering models and theories.

Applications to Systems

We anticipate that formal (computational-logic-based) methods have key roles to play in at least two major areas. One is in modeling, design, analysis, and assurance of high-assurance *systems* themselves. This work would place *model-based systems engineering* on a much sounder foundation.

A broad range of formal methods, each rooted in specific mathematical structures, and related notations analysis methods, and each narrow in scope, as a tradeoff to obtain automated analytics within that scope, will be most relevant in this area. Examples include *temporal logic* and *model checking* (Clarke, 1986), methods based on *satisfiability modulo theories* (SMT) solvers (deMoura, 2011), hybrid automata (Alur, 1993), dynamic fault trees (Sullivan, 1999), and other general-purpose, logic-based tools and methods.

Applications to Theory Development

The second area in which formal methods have an important role to play is in enabling and promoting increased rigor in the development, validation, presentation, and automation of new models and theories of systems engineering. In this space, more expressive formal notations and underlying math-logic frameworks (e.g., higher-order logic) are likely to provide the greatest power, even at the cost of some loss of automation.

Examples include the use of higher-order constructive logic and related proof assistants, such as Coq (Chlipala, 2013; Coq, 2014; sf, 2015), PVS (pvs, Owre, 1992), Agda (Bove, 2009), as well as the emerging class of practical programming languages based on dependent type theory, such as Idris (Brady, 2013) and F* (Swamy, 2015). These languages provide incredible levels of expressiveness, and the unification of proof theory and programming. They enable the specification of demanding computational properties and the rigorous verification of such properties within the same *programming-and-logic* languages.

5.1.2 THE HUMAN DIMENSION

The complex systems of the future will not only serve human ends, but will have to integrate deeply with human and social phenomena. People and organizations are computational and physical elements within larger cyber-physical-human systems. The reality of deepening cyber-human integration and the fundamental research challenges it poses were recognized in the August, 2015 "PCAST Report" to the President (pcast, 2015).

While we are not proposing a deep dive into applied cognitive or social science, or the use of people to carry out complex or large-scale computations, the work that we are proposing will provide a rigorous approach for developing, validating, and expressing theories and models in these areas, in ways that are useful to the systems engineering enterprise. More specifically, our frameworks link means-ends hierarchies of *technical* system qualities to individual stakeholder quality requirements and their underlying subjective value propositions. Our proposed and ongoing work thus links the technical dimensions of system engineering to value-based systems engineering theory (Boehm, 2007).

5.1.3 DOCUMENT ORGANIZATION

In the rest of document, we summarize preliminary views, plans, and results that we have developed to this point. These views will be tested and will evolve and be extended by a workshop that we are planning to run as part of this seedling project in September, 2015. This document can be read as a preliminary proposal for a significant research and development project in this area. The end goals of such research, in turn, would be to strengthen both the theoretical and the practical (technological and methodological) foundations of systems engineering. Such work is now critical to the health of the field and to the success of a broad range of critical, complex, current and future, software-intensive cyber-physical-human systems.

5.2 Opportunities and Challenges

Systems engineering is at a crossroads. It evolved mainly to enable more effective development of physical-human systems, predominantly for defense applications. It is now being disrupted profoundly by the ongoing revolution in computer and information science and engineering (CISE). This revolution is leading us rapidly into an era of unprecedented, software-intensive, cyber-physical-human systems. The discipline of systems engineering is not well equipped today to handle the now-dominant *computational* aspects of modern systems. Nor are the CISE disciplines well prepared to manage the broader, non-CISE aspects of complex systems, including complex physical and human/social phenomena.

At the same time, the scope of systems engineering is expanding, and it should and must expand, to a broad range of new domains, including health and healthcare, transportation, smart cities, cyber-security, and critical infrastructure, among others.

Both defense and these other domains are critical to our society. The revolution in computing creates tremendous opportunities. The computer science fields alone are not adequately equipped to fully exploit these opportunities. Systems engineering has a vital role to play. But the changes in the landscape are straining the field and its capabilities to a considerable degree.

5.2.1 Unprecedented Potential

The most important driver of these stresses is the change in the nature of systems. From physical-human systems, we are now entering an era of computationally automated, self-creating, resilient, rapid learning, cyber-physical-human (CPH) systems.

Such systems are now increasingly global in scale, integrated into ever larger systems of systems, with the capacity for trans-human perceptual capabilities based on massive data analytics. They will increasingly combine inductive learning from data at ultra-scale, deductive reasoning, and automated, ongoing multi-armed experiments to learn and to act on what is learned so as to evolve in highly automated fashion to higher states of fitness for purpose within their co-evolving environments.

Systems are transitioning from being merely *allopoietic*, which is to say that they produce something for *others*, to being *autopoietic*, which is to say that they will be producing *themselves* in an ongoing manner (Gabriel, 2006). Humans will be integral and will be in (parts of) “the loop”, but future systems will no longer rely entirely on people to perceive threat, needs, and opportunities, and evolve to meet them. Indeed, as our colleague, Barry Horowitz has observed (personal communication), systems are transitioning from being human-run and semi-automated to being computer-run and semi-manual, with automated processes delegating to people only to handle tasks that machines are not yet capable of handling effectively. We are genuinely entering uncharted territory.

5.2.2 Unprecedented Challenges

At the same time, this vision is met by problems that make it extraordinarily difficult, costly, and risky to build not only the next generation but even the current generation of cyber-physical systems (e.g., advanced aircraft), or cyber-human systems (c.f., the recent failures in cyber-human systems security at the U.S. Office of Personnel Management (opm, 2015)). We are facing critical gaps in capabilities in major areas, including “requirements,” the ability to achieve certain requirements, the ability to make sensible tradeoffs among them, and systemic, overall systems *value assurance*.

In the area of requirements, even determining what to build, which must include articulating all key system qualities in all dimensions, remains beyond the state of the art. In the area of systemic assurance of system value - which generalizes the notion of safety assurance to the ultimate goal of value creation net of costs and risks - is also beyond either our theoretical or practical abilities. For example, we lack an adequate understanding of how to integrate *evidence* (e.g., from simulations, testing, or analysis of models), with *deductive reasoning*, to derive bounds on the confidence we can justifiably have in projects and systems.

5.3 PROBLEM

The problem that we intend to address is that the systematic design and development of the systems of the future, and even of many of today's most demanding systems, with acceptable assurance of success, is infeasible with current theory, methods, and tools. The result is that we are facing daunting and unacceptable risks and outright failures, including cost and schedule overruns, under-performance, loss of strategic data, and physical failures, in critical systems engineering projects and in operational systems.

5.4 NEEDS

In the area of requirements, we face some fundamental challenges. The environments and systems we need are inherently complex. We need ways of managing this complexity. One approach is through modeling and analysis. Yet domain complexity dictates the inevitable need to model heterogeneous sub-domains, for widely varying technical qualities, and thus with diverse forms of modeling and analysis.

At the same time, the results of individual analyses, constituting heterogeneous forms of *model-derived evidence*, will have to be integrated and evaluated in an ongoing manner to support higher-level reasoning and judgments about system qualities and projects value, and about appropriate adjustments and courses of action. The goal is to enable evidence-based analysis and decision-making that accounts for all critical system parameters (requirements, designs, etc.).

Yet what projects typically face today is a pervasive lack of rigor in many dimensions. These include analysis of value propositions; precise definition, specification, and assurance of system qualities; in the notations used in documenting requirements, designs, qualities, and decisions; and in the manner of reasoning applied to such artifacts and in decision making.

At the end of the day, and in the context of the kinds of systems that are now needed, and in some cases being built, we have little basis for justifiable confidence that system and project value is reasonably assured. This state of affairs generally holds from the early stages of the system life-cycle through systems operation and evolution. Even systems that appear highly successful, such as civilian aviation, are exhibiting unacceptable weaknesses in key dimensions, as evinced, for example, by the Government Accountability Office's recent report on systematic cyber-security weaknesses in the National Airspace System (again a cyber-driven dimension of uncontrolled complexity).

What we need now are new approaches to systemic value assurance, integrating assurance of individual and composite technical system properties (schedulability, availability, security, usability, etc.) with validated stakeholder propositions to support judgments of the form *with acceptable residual risk, all success-critical stakeholders can be satisfied with the system: as is or as it is being developed*. The current state of the art does not provide a means for managing technical and stakeholder value and risks to this degree. In particular, we do not adequately track evolving requirements, parameterized in all key

dimensions (e.g., individualized stakeholder needs, operational contexts, system states, development phases, etc); nor do we adequately characterize linkages between technical properties and stakeholder value propositions. We certainly do not adequately specify and assess *evidence of satisfaction* of the full range of critical system technical properties modulo acceptable remaining epistemic gaps and known risks.

We aim to understand and validate a framework to enable such reasoning so as to inform decision-making across the systems lifecycle. The claim is *not* that we can eliminate risks and epistemic gaps. That is clearly not going to be possible. Rather, we need an evidence-based framework within which we can reduce them to a practical extent and within which we can characterize what gaps remain, so that we can make informed human judgments about courses of action: e.g., to cancel a project, proceed, intervene, deploy, etc.

The evidence and reasoning that support such judgments should be clear, based on sound theory, and validated. It should address all relevant technical system properties (ranging from physical to cyber to cognitive and social). Evidence can come by deduction from system models (e.g., proofs of correctness of particular software elements, or deductive proofs that certain designs have certain properties, albeit with remaining model risks), from test and evaluation, from human engineering judgments, from scientific studies, etc. In particular, we expect significant forms of evidence to come from the use of formal methods and associated tools applied in the context of model-based systems engineering.

Such evidence should ultimately address technical requirements and key stakeholder needs. It should enable principled and transparent decision-making regarding technical and value tradeoffs. It should be updated and validated throughout the lifecycle. It should be made reliable and efficient with advanced support from software tools and environments, including integrated formal methods suites and methods for combining inductive reasoning from data/evidence (e.g., from model-based analysis) with logical deduction. And all of this should be validated by theory and usable in practice at scale.

5.5 APPROACH

It is clear that no single project or advance will solve the problems we have articulated. Yet carefully designed, innovative, approaches can significantly improve the situation by creating new vectors for the evolution of the field. We propose to do so with a project that is intended to integrate a *diversity* of cutting-edge formal methods into systems engineering.

Individual formal notations and analysis methods have, of course, been used for years to support reasoning in certain narrow areas. Examples include the use of hybrid automata and related analytics for reasoning about interactions of discreet (computational) and analog (continuous physical) phenomena. There are many examples, ranging from the use of carefully developed physics models to fault tree reliability analysis based probability theory, Boolean logic, binary decision diagrams, Markov chains, etc.

5.5.1 AN *INTEGRATIVE* APPROACH

What distinguishes the approach we propose is an emphasis on a degree of comprehensiveness, integration, and broad *expressiveness* that is needed to address the real *systems* problem. As Erik Herzog of SAAB stated at a recent workshop, “Success does not come from excellence in a single property, but by excelling in the combination of properties (Herzog, 2015).” Now is the time to focus on the integration of formal methods in order to address the *integrated diversity* of concerns that must be balanced in modern systems.

Specific technical constraints on the integrated approach we propose to explore include the following. It should:

- address both technical and stakeholder value concerns
- be applicable to diverse system and application domain models
- address the full means-ends hierarchies of systems properties
- be parameterizable by property-specific formal languages
- support variation by stakeholders, states, and evolutionary phase
- exploit property assurance hierarchies to inform value judgments
- accommodate diverse underlying formal notations and analytics
- accommodate heterogeneous system models in integrated fashion
- support proof engineering with automated decision procedures
- provide precise property definition, specification, and assurance
- provide modelers with great flexible in degree of detail modeled
- support formal notations capable of expressing arbitrary mathematics
- provide a framework for integration/analysis of evidence for assurance

The approach we propose thus integrates across multiple dimensions: of technical properties and stakeholder value; across formal underpinnings, notations, and analysis methods; across the broad range of critical system properties that ultimately contribute to value and satisfactory systems; and across stakeholder preferences, system and environment states, and development phases. We have already made considerable progress on this vision. The following diagram indicates the scope. We envision an integrated approach to specification, modeling, analysis, and ultimately evidence-based assurance for hierarchies of system properties, leading to value creation, as illustrated here. The tree shown in Figure 5.1 is *definitional;* it provides a structure for organizing requirements; and it is a proof tree for assurance.

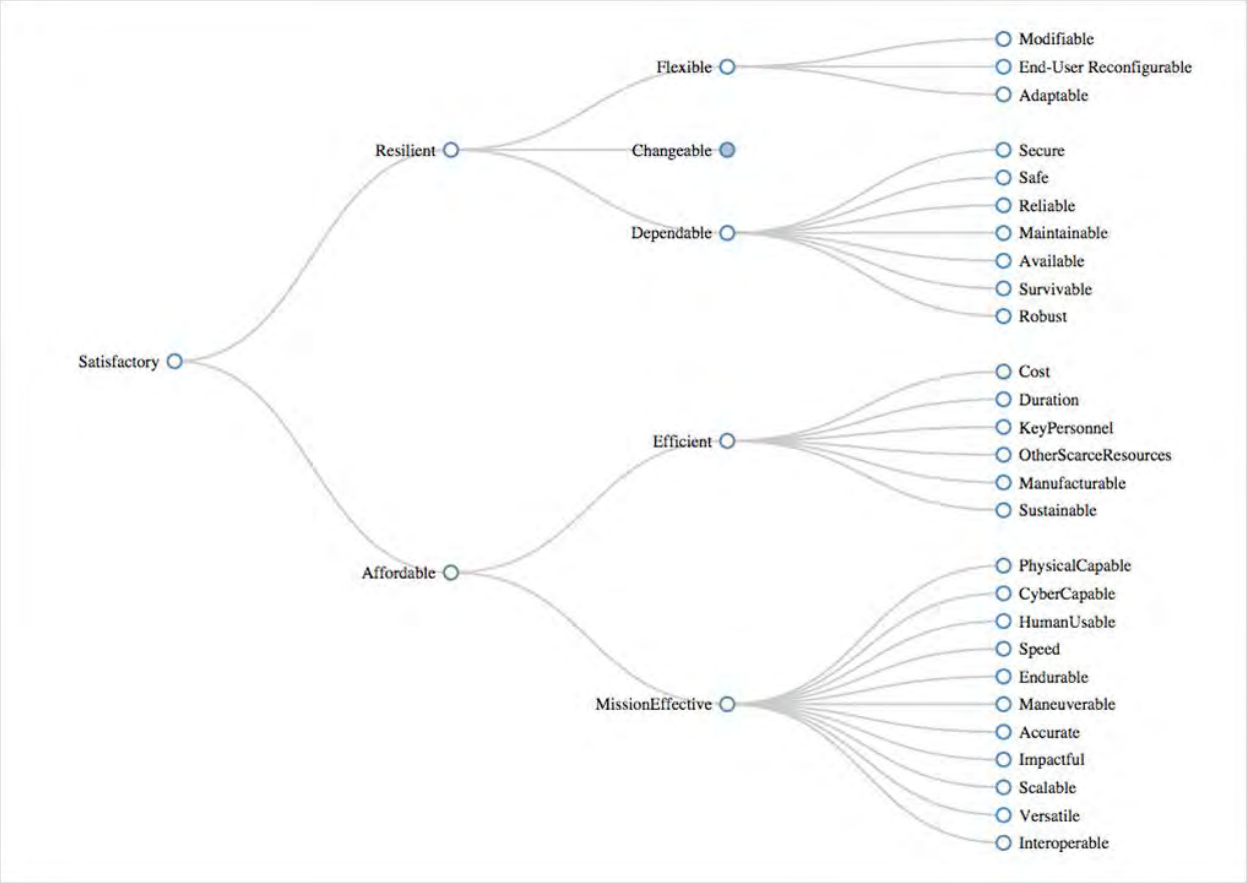


Figure 5.1. Definition, Specification, and Constructive Logic Proof Tree for Systemic Value Assurance

5.5.2 Model-Based Systems Engineering

Co-PI John Baras's current efforts in model-based systems engineering are broadly consistent with and supportive of the agenda articulated here. His current work includes efforts to develop an integrative and formal approach to *model-based systems engineering*.

He is developing an integrated modeling hub built around SysML. It incorporates meta-modeling methods and environments, and interfaces with a variety of domain specific design methods and tools, including support design space exploration and tradeoff analysis. It achieves important degrees of flexibility and expressiveness through links with parametric and requirements diagrams of SysML and with multi-criteria mixed optimization and constrained based reasoning tools. It directly addresses the need for new approaches to representing and managing complex, multi-dimensional systems requirements. It integrates a range of formal methods based on model checking, contract based design, theorem proving, and finite time temporal logic specifications.

5.5.3 Systems Quality and Value Assurance

Co-PI Kevin Sullivan's current efforts to produce a precise ontology of system qualities and stakeholder value that supports quality definition, specification, and assurance in a manner that can be adapted to the details and needs of arbitrary domains, is also highly supportive of the proposed agenda.

With his colleague, Barry Boehm, and their students, he is developing a formal, integrated theory of system qualities and stakeholder value expressed in the higher-order polymorphic computational (constructive) logic of the Coq proof assistant. The work formalizes and broadly generalizes Boehm's quality ontology (Boehm, 2015). It provides an integrated treatment of value, properties, strategies, and tradeoffs. It is parameterized by domain and property-specific specification and modeling languages.

A notable characteristic of the approach is that it is designed to accommodate a broad and evolving family of property-specific modeling languages and related analytics. The idea is that a comprehensive system specification will have to be articulated in different languages for different properties; different analyzers will be needed to provide assurance for different properties; and a unified approach is needed to folding evidence from these low-level analyses into conclusions at higher levels of reasoning and assurance.

Many of the required languages do not yet exist. As a case study in the development of new and improved property-specific languages and analytics (such as decision procedures), Sullivan et al. (Sullivan, 2015) formalized and extended and improved Ross's "semantic basis" approach (Ross, 2015) to specifying a diverse set of kinds of *changeability* properties. Sullivan et al. have (work in progress) developed two separate languages in this regard: one for expressing pre-condition and post-condition-based concepts of changeability, and the second formalizing a framework for modeling modularity in system design and the flexibility it affords.

This work substitutes formal definitions for the endless debates over vague and informal definitions of system properties that have plagued the field for decades. The work does not purport to provide any kind of definitive resolution. Rather, it enables one to bring preferred *concepts* to the table and have languages developed in which those particular concepts (e.g., of resilience) can be not only formalized but integrated into an overall framework for property definition, specification, tradeoff analysis, and evidence-based assurance.

5.6 Evidence that it Will Work

There are good reasons to believe that the approach that we are pursuing has the potential to produce *significant* advances in the foundations of systems engineering. First, it addresses the software problem, but in the systems context. Second, it leverages rapid and ongoing advances in formal methods. Third, there is a natural and compelling opportunity to integrate Baras's work on integrated tool suites with Sullivan's work on property hierarchies and value.

5.6.1 Addresses the Software Problem

First, as we have noted, systems are now deeply computational, and software is now the preferred "building material" for complex systems. It not only integrates the parts but animates the core functionalities of complex systems, from jets to mobile computing and communications networks to healthcare delivery systems.

The problem is that *software* development remains fraught with great difficulties, particularly when it comes to high assurance systems and the integration of software and computation with complex physical and human/social phenomena. The software-intensive cyber-physical-human (CPH) systems of the future thus place demands on *software*, in the complex systems context, that neither systems nor software engineering fields can fully meet.

The problems span the gamut of systems engineering issues discussed above, but in the software-specific arena, including requirements that address the full range of system technical qualities and stakeholder value objectives; architectural design; verification; evidence-based assurance; and, especially perhaps, system evolution.

The approach we are proposing directly addresses the software problem: not as a *pure software* problem, but as one facing software-intensive CPH *systems engineering*. We are therefore not only interested in the properties and value of *software* but in the properties and value at the overall systems level. It is the systems that count, yet software animates them.

The opportunity is to lift important foundational work in software engineering to the systems engineering level. This is happening, of course, in various pockets, e.g., as seen in work on hybrid automata in the cyber-physical-systems community. But to our knowledge there is no comprehensive and *integrative* program in this regard.

We believe that is what is now needed. Addressing the full range of relevant properties, which is what is needed, requires integration of *diverse* forms of domains, models, notation, analysis techniques, and evidence. To the extent that progress is already occurring within narrow vertical sub-areas, we will leverage it. What we offer, rather, is not only progress in narrow areas or individual "ilities", but at the integrative, *systems* (and ultimately *value*) engineering level.

5.6.2 Leverages Advances in Formal Methods

The concept of formal methods began in the mid-20th century with the early (1960s and 1970s) work of pioneers such as Floyd, Hoare, Dijkstra and others. Such work gave rise to set-theoretic formal specification languages, such as Spivey's *Z* (Spivey, 1989), enabling scalable formal modeling, specification, and verification of critical software components of significant real-world systems, albeit with significant manual effort (Wing, 1990).

The discovery of efficient and scalable approaches to temporal logic model checking by Clarke and others (Clarke, 1986) dramatically advanced the value of formal methods by enabling fully automated verification of constrained classes of programs against specifications of behavioral properties expressed in various temporal logics.

Research on the extension of such techniques to a broader range of software represented another significant advance. For example, the work of Ball, Rajamani (Ball, 2011), and others demonstrated that model checking could be applied to abstractions of software to check that clients behave properly with respect to the usage rules of given interfaces. This work led directly to all but complete elimination of Microsoft Windows operating system's "blue screens of death," most of which were caused by mis-behaving third party device drivers running in the privileged operating system context.

Today we are seeing the integration of limited but fully automated theorem checking engines into software compilers and other tools, to enable static checking of an increasingly broad range of technical system properties, such as assurance that nil pointers will not be dereferenced. Much of this work, in turn, relies on formal methods rooted in *satisfiability modulo theories* solvers, such as Z3 (deMoura, 2008; deMoura, 2011) - programs that implement decision procedures for propositions in constrained formal domains, such as those of bit vectors, Peano arithmetic, and so on.

Finally, we are now seeing important and impressive advances being made in highly expressive formal methods based not on set theory but on *dependent type theory*. This work unifies computation involving pure functions with higher-order logic and natural deduction, which allows for programs, propositions about programs, and proofs of such propositions to be expressed and checked all in the same expressive languages.

Dependent type theory (Martin-Löf, 1984; Altenkirch, 2005) was invented to help establish the logical foundations of constructive mathematics. Today it is being used not only for the production of certified programs (see work by Chlipala (Chlipala, 2013) and Morrissett (Morrissett, 2012), for example), but also by research mathematicians, who have use it to formalize a vast array of abstractions that we believe are relevant to expressing dynamical properties of physical systems. See, for example, the work of Fields Medalist Voevodksy on homotopy type theory (Hott, 2013).

We are finding such notations to be incredibly useful in expressing broad systems engineering models of mean-ends property hierarchies and value in ways that can be parameterized by arbitrary, domain-specific systems models, and in the design and *integration* of diverse formal languages specialized to support the definition, formal specification, and assurance of particular properties and even particular models of such properties. (E.g., different systems might need different models of, and different languages for specifying, *resiliency* properties.)

The bottom line is that the field of formal methods is now advancing rapidly, and beyond the stage at which it is relevant mostly to software and software properties. Trends in formal modeling and fully automated verification, as well as in practical applications of type theoretical tools for system and property description, language design, and proof engineering as a form of *evidence integration,* all have direct relevance now to systems engineering.

5.6.3 Combines Tool Integration with Model of Multi-Property Assurance

Of particular interest to us is the opportunity we now have to connect Baras's work on an integrate tool bench with Sullivan's work on models of property hierarchies. This combination has the potential to support the integration of evidence, produced by diverse modeling and analysis tools, with the system property and value ontologies and deductive proof tree reasoning techniques developed by Sullivan in collaboration with Boehm. The approach would be consistent with the findings of a report (in draft) by Rushby (Rushby, 2015).

Dr. Chris Paredis of the National Science Foundation nicely articulated the opportunity we have. He said that The fact that we express information and knowledge more formally is likely to lead to entirely new processes, methods, and tools that make extensive use of computer-supported reasoning, analysis and optimization. To best take advantage of this opportunity, we need to build on a rigorous foundation (Paredis, 2015). ”

5.7 Who Will Care?

Success in the effort that we are proposing would be of significant value to a broad array of stakeholders. These include the systems engineering research and practitioner communities; funding agencies; systems engineering, computer science and engineering educators and students; industrial organizations, and government. The intellectual merit and potential for broader impact are both high.

5.8 What Will it Cost?

Cognizant of constraints on SERC funding streams, we nevertheless envision a three to five year effort with a team of anywhere from two to perhaps ten or so investigators. We model this vision based on the scope of current RTs, such as RT-113.

Investigators would be expected to have either backgrounds in formal methods or in content areas that most need to be formalized. Funding should be adequate to support the education of an appropriate number of Ph.D. students and perhaps postdocs.

5.9 How Will We Know We're Succeeding?

We will run a small workshop on this topic in September 2015, to build an initial research community and to help validate planned directions. Within one year of funding we would expect to produce a minimal viable demonstration system and research publications on underlying conceptual advances. Within three years we would aim to produce and validate strong theoretical foundations, and prototype systems capable of being demonstrated on realistic problems. Within five years we would expect to see significant validation of both our theory and tools in pilot applications at realistic levels of scale and complexity. We would expect to publish research papers in top systems engineering and related conferences and journals on an annual basis starting in year one.

5.10 Conclusion

Today systems engineering stands on shaky foundations. This project will strengthen them. It will establish formal mathematical and related technological foundations, with strong potential for powerful impact on practice, and in ways that are critical for the health and effectiveness of the field.

References

(Altenkirch, 2005) Altenkirch, T., McBride, C., McKinna, J., Why dependent types matter, (April 2005), PDF: http://www.cs.nott.ac.uk/~txa/publ/ydtm.pdf.

(Alur, 1993) Alur, R., et al. *Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems*. Springer Berlin Heidelberg, 1993.

(Ball, 2011) Thomas Ball, Vladimir Levin, and Sriram K. Rajamani. 2011. A decade of software model checking with SLAM. *Commun. ACM* 54, 7 (July 2011), 68-76.

(Boehm, 2007) Boehm, B. andJ. Apurva, The value-based theory of systems engineering: identifying and explaining dependencies, INCOSE International Symposium, 2007.

(Boehm, 2015) Boehm, B. and N. Kukreja, An initial ontology for system quality attributes, 25th Annual INCOSE International Symposium (IS2015), July 13--16, 2015.

(Bove, 2009) Bove, A., Dybjer, P., and U. Norell, A brief overview of Agda--A functional language with dependent types, Theorem Proving in Higher-Order Logics, Vol. 5674, (2009), LNCS, pp. 73--78.

(Brady, 2013) Brady, E., Idris, a general-purpose dependently types programming language: design and implementation, Journal of Functional Programming 23.05, (2013), pp. 552--593

(Chlipala, 2013) Chlipala, A., Certified Programming with Dependent Types: A Pragmatic Introduction to the Coq Proof Assistant, MIT Press, Dec 6, 2013.

(Clarke, 1986) Clarke, E.M., et al., Automatic verification of finite-state concurrent systems using temporal logic specifications, ACM Transactions on Programming Languages and Systems, 8:2, April 1986, pp. 244-263.

(Coq, 2014) Coq Dev. Team, Reference Manual (Ver. 8.4pl6), Inria, 2014, https://coq.inria.fr/distrib/current/refman/

(deMoura, 2008) de Moura, L. and N Bjørner, Z3: An efficient SMT Solver, Tools and Algorithms for the Construction and Analysis of Systems, Lectures Notes in Computer Science Vol. 4963, 2008, pp. 337--340.

(deMoura, 2011) De Moura, L. and N. Bjørner. Satisfiability modulo theories: introduction and applications. *Communications of the ACM* 54.9 (2011): 69-77.

(Gabriel, 2006) Gabriel, Richard P., and Ron Goldman. "Conscientious software." *Acm Sigplan Notices*. Vol. 41. No. 10. ACM, 2006.

(Herzog, 2015) Herzog, E., SAAB, Searching for a Partner in .SE, Abstract for LCCC-ACCESS workshop on Model-Based Engineering, Lund University Center for Control of Complex Engineering Systems, May 2015, http://www.lccc.lth.se/index.php?mact=ReglerSeminars,cntnt01,abstractbio,0&cntnt01abstractID=729&cntnt01returnid=116.

(Hott, 2013) *The Univalent Foundations Program,* Homotopy Type Theory: Univalent Foundations of Mathematics, Institute for Advanced Study, 2013. Available at: http://homotopytypetheory.org/book.

(Martin-Löf, 1984) Martin-Löf, P. Intuitionistic Type Theory, Bibliopolis. PDF available at: http://www.cs.cmu.edu/afs/cs/Web/People/crary/819-f09/Martin-Lof80.pdf.

(Morrisett, 2012) Greg Morrisett, Gang Tan, Joseph Tassarotti, Jean-Baptiste Tristan, and Edward Gan. 2012. RockSalt: better, faster, stronger SFI for the x86. *SIGPLAN Not.* 47, 6 (June 2012), 395-404.

(opm, 2015) Harsh Truths from OPM Hack: More Monitoring is Coming, U.S. News & World Report, July 23, 2015.

(Owre, 1992) Owre, S., Rushby, J.M., and N. Shankar, PVS: A prototype verification system, Automated Deducation, CADE-11, Lecture Notes in Computer Science Vol. 607, pp. 748-752.

(Paredis, 2015) Paredis, C., Research Directions for Developing a Rigorous Foundation for MBSE, Abstract LCCC-ACCESS workshop on Model-Based Engineering, Lund University Center for Control of Complex Engineering Systems, May 2015, http://www.lccc.lth.se/index.php?mact=ReglerSeminars,cntnt01,abstractbio,0&cntnt01abstractID=739&cntnt01returnid=116.

(pcast, 2015) President's Council of Advisors on Science and Technology, Report to the President and Congress: Ensuring Leadership in Federally Funded Research and Development in Information Technology, August 2015.

(pvs) PVS Verification System (Version 6.0). SRI, http://pvs.csl.sri.com/.

(Ross, 2015) Ross, A. and D.H. Rhodes, Towards a prescriptive semantic basis for change-type ilities, Procedia Computer Science, Vol. 44, 2015 Conference on Systems Engineering Research (CSER 2015), March 17-19, 2015, Hoboken, NJ, USA, pp. 443--453.

(Rushby, 2015) forthcoming technical report on understanding assurance cases (personal communication).

(Pierce, 2015) Pierce, et al., Software Foundations, Jan., 2015, http://www.cis.upenn.edu/~bcpierce/sf/current/index.html

(Spivey, 1989) Spivey, J.M., The Z notation: a reference manual, Prentice-Hall, Inc. Upper Saddle River, NJ, USA, 1989.

(Sullivan, 1999) Sullivan, K. Bechta Dugan, J., and D. Coppit, The Galileo fault tree analysis tool, Fault-Tolerant Computing Systems, 15--18 June, 1999, pp. 232-235.

(Sullivan, 2015) Ke Dou, Xi Wang, Chong Tang, Adam Ross and Kevin Sullivan, "An Evolutionary Theory-Systems Approach to a Science of the Ilities," Procedia Computer Science, Vol. 44, 2015 Conference on Systems Engineering Research (CSER 2015), March 17-19, 2015, Hoboken, NJ, USA

(Swamy, 2015) Swamy, N. et al., Dependent types and multi-monadic effects in F*, Microsoft Research, Inria, MSR-Inria, University of Maryland, ENS Paris, IMDEA Software Institute, Draft, available at https://www.fstar-lang.org/papers/mumon/. For more information, see https://www.fstar-lang.org.

(Wing, 1990) Jeannette M. Wing. 1990. A Specifier's Introduction to Formal Methods. *Computer* 23, 9 (September 1990), 8-23.

6 Policies and Practices for Model-Centric Government-Industry Collaborative Environments for Systems Engineering and Development – Gary Witus, Wayne State University

6.1 Objectives

Our broad objectives are to develop Systems Engineering policies, practices, methods and tools to advance the twin Department of Defense (DoD) initiatives (1) to "Own the Technical Baseline," and (2) to transition to "Digital Engineering Design." These initiatives are part of DoD's response to the 2009 Weapon System Acquisition Reform Act (WARSA) which calls for competitive prototyping, dual-sourcing, use of modular, open architectures to enable competition for upgrades, use of build-to-print approaches to enable production through multiple sources, acquisition of complete technical data packages, periodic competitions for subsystem upgrades, and licensing of additional suppliers.

The Deputy Assistant Secretary of Defense (DASD) Digital Engineering Design vision is of a "Digital Thread" across development lifecycle accompanied by a "Digital Twin," both based on an underlying "Digital System Model" template. The Digital Thread, Digital Twin, and Digital System Model are a vision of the Technical Baseline for the digital age. All of the Service branches have initiatives to begin to develop an implementation of the Digital Thread and Digital Twin, including pilot testing partial implementations. Digital Engineering Design necessitates model-sharing and model-exchange between the US Government (USG) and Defense Industry. Achieving the necessary open access to the models and data will require data rights management to protect and/or compensate for Trade Secrets and/or Intellectual Property (IP) that are exposed in the collaborative digital environment.

> *Digital Thread: "An extensible, configurable and component enterprise-level analytical framework that seamlessly expedites the controlled interplay of authoritative technical data, software, information, and knowledge in the enterprise data-information-knowledge systems, based on the Digital System Model template, to inform decision makers throughout a system's life cycle by providing the capability to access, integrate and transform disparate data into actionable information."*
>
> *Digital Twin: "An integrated multiphysics, multiscale, probabilistic simulation of an as-built system, enabled by Digital Thread, that uses the best available models, sensor information, and input data to mirror and predict activities/performance over the life of its corresponding physical twin."*

Digital System Model: "A digital representation of a defense system, generated by all stakeholders that integrates the authoritative technical data and associated artifacts which define all aspects of the system for the specific activities throughout the system lifecycle."

Our focus and objectives are designed to complement and leverage related initiatives across DoD and the Defense Industry. During the initial investigation, we conducted detailed interviews with stakeholders representing different perspectives across DoD. The purpose of the interviews was to understand related programs and initiatives, gaps and priorities, near- and long-term challenges in order to formulate our objectives The DoD stakeholders we interviewed included:

- Philomena Zimmerman, Scott Lucero and Tyesia Alexander (DASD; Digital Engineering Design and Modular Open System Architectures)
- Dave Gorsich and Curt Adams (TARDEC; the Chief Scientist and the lead for the TARDEC Digital Thread)
- Howard Owens and Brent Gordon (NAVAIR; Technical Data Package Improvement and Lifecycle Management)
- Kevin Massy (DARPA, Tactical Technology Office Program Manager for the META and AVM programs)
- Randall Gaereminck (TARDEC; Associate Director for Product Lifecycle Engineering and Product Data Lifecycle Management, and TARDEC lead in the RDECOM/AMRDEC Net Centric Model Based Enterprise Phase II)
- David French (Warner Robbins AFB; Reverse Engineering and Re-Engineering lead)
- Col. Timothy West (USAF; Arnold AFB engineering development)

We combined the insights from these interviews with findings from our literature review, and findings from previous, extensive interviews with Government agencies and Industry on collaborative model centric engineering technical approaches (conducted under RT118/141) to formulate our specific objectives and technical approach.

The project's focus is on challenges to "Owning the Technical Baseline" and implementing Digital Engineering Design related to model & data specification, data rights issues, collaboration practices and procedures in a competitive acquisition environment. Some of the significant issues and gaps in current knowledge and practice identified during the initial investigation include the following:

- Evidence to convince Program Managers (PMs) of the cost-vs-value of acquiring data rights to the Technical Data Package (TDP) realized as the Digital Thread and Digital Twin
    - The value of competition and second sourcing
    - The value of accelerated development and reduced defects
    - The value over the operational lifespan given the reality of Diminishing Manufacturing Sources and Material Supplies (DMSMS)
- Understanding the data rights issues in different elements of the Digital Thread and Digital Twin, and methods to determine what the rights "should cost" balancing USG and vendor chain perspectives

- Understanding what to specify for delivery and how to specify Data Item Descriptions (DIDs) and Contract Data Reporting Lists (CDRLs) for fully digital artifacts to create the Digital Thread and Digital Twin, and to "Own the Technical Baseline"
- Commercial technical data, models, and data rights belonging to vendors
- Vendor initiatives in model-centric engineering enterprise
- Data rights management policies and practices matched to Digital Engineering Design collaboration standards to address potential Trade Secret and Intellectual Property exposure
- Entrenched acquisition practices, and guidelines for cost-effective incremental transition steps
- Verification and Validation (V&V) of the Digital Thread / Digital Twin, and V&V of the Digital Thread / Digital Twin guidelines and specifications
- Process and product compatibility, and the impacts of Information Technology (IT) compatibility, during the transition from the "as is" system to the envisioned Digital Engineering Design

The long term objectives of the project are:

A. To develop a reference model for the Digital Thread and Digital Twin over the lifecycle. When completed, the reference model will identify what Digital Thread and Digital Twin features and capabilities that are needed at what points during the lifecycle, what deliverable to specify for the Digital Thread and Digital Twin via DIDs and CDRLS, and an enterprise data rights framework for collaborative Digital Engineering Design for model exchange and data sharing in the Systems Engineering workflow. The enterprise data rights framework manages which parties have access to which portions of the Digital Thread / Digital Twin, and the type of access, to implement the collaboration and data rights agreements. The reference model includes a model of the different types of access, e.g., copy, read, write, indirect reference, execute, etc., and the associated data rights for the type of data and type of access. Essentially, this specifies transformations and viewpoints open to different parties at different stages of the acquisition lifecycle, and transformations between them.

B. To supplement the reference model with information to enhance its usefulness for (1) transition strategy planning, and (2) justifying the costs associated with the digital TDP to PMs. The supplemental information will identify near-, mid-, and long-term issues and gaps in the transition to Digital Engineering Design. It will identify existing or emerging potential solutions, standards and approaches and risks. It will, to the extent possible, document costs, savings, and other benefits at different points in the model, including the value of "Owning the Technical Baseline" (and costs of not owning it), the impacts of competition, what the digital TDP "should cost" including management of and compensation for Trade Secret and IP exposure.

C. To develop a strategic planning tool for orderly incremental transition from current practices to the vision of "Owning the Technical Baseline" and collaborative Digital Engineering Design. Transition from current practices to Digital Engineering Design will be incremental and piecemeal, as seen in the varying stage and scope of the pilot programs. The planning tool will help identify high-value, low-risk stages that will link increasingly complete chains.

6.2 CURRENT STATE OF PRACTICE AND LIMITATIONS

USG/Industry collaborative Digital Engineering Design in a competitive acquisition environment involves many open and complex issues. These issues include understanding what technical content needs to be in the Digital Thread/Digital Twin at different stages of acquisition and development processes, transition strategies from the current system, what technical data must be exposed, what type of data rights are needed by whom and when, how which data rights should be compensated, the value of openness, competition, and digital continuity vice the costs of data rights, and how to control and manage access to technical data in the collaborative engineering environment.

Interviews with NAVAIR provided relevant insights. They stressed the need for a reference model for the digital thread that supports implementation approaches that iteratively refine digital system model over the acquisition lifecycle, as well as strategies that transform or translate the digital system model across development stages. They stressed the need for a reference model in which the needs later in the acquisition lifecycle (e.g., O&S spares and upgrades) are anticipated at the front end of the digital thread. This would reduce risks from oversights, and facilitate the downstream process. They observed that access to the digital TDP as it is being developed has potential to reduce program risk because it provides greater visibility into technical progress. They emphasized the need to integrate manufacturability concepts and constraints into the acquisition efforts, as well as concepts for operations and sustainment, and contracts.

The current state of practice is evolving on many fronts. Policies, practices and templates being developed and piloted at the DASD(SE) and within the Service branches. All of the Service branches have initiatives to plan, develop and pilot implementations of the Digital Thread and Digital Twin. Commercial standards for data exchange in the Digital Thread (e.g., the ISO Standard for the Exchange of Product model data - STEP interoperability standard) and for model-exchange in the Digital Twin (e.g., Functional Mockup Interface – FMI – standard) have been developed and are being piloted. The Defense Industry majors – Boeing, Lockheed Martin, Northrop Grumman, Raytheon, etc. – and the automotive industry are implementing the methods and technologies of the Digital Thread /Digital Twin, and collaborative engineering environments internally, as are some USG agencies such as NASA/JPL. They are demonstrating feasibility and value, but fall short of full system development with the Digital Thread/Digital Twin.

NIST and OSD have been hosting an annual summit on “Model-based Enterprise and Technical Data Package” since 2009. This is the premier forum in which Defense agencies and Defense Industry meet to share the latest technological practices for model based engineering and model-based technical data packages.

While there has been considerable progress in identifying and solving technical issues in the Digital Thread, there are still technical challenges in extending the Digital Thread to a broad range of manufacturing-related activities such as assembly, bidding, engineering changes, in-process inspection, etc.

There has also been considerable progress in solving the technical issues in physics-based high-fidelity computational modeling for the Digital Twin, e.g., CREATE-Ship and CREATE-AV. Technical issues remain, especially with regard to high-energy events (e.g., blast), behaviors (software), wear- corrosion- and fatigue-induced changes in material microstructure and mechanical properties, manufacturing defects, and predictions under conditions with unknown boundary conditions (e.g., terrain), extending the models to include the effects of manufacturing and finishing processes. High resolution models,

such as CREATE-Ship and CREATE-AV, require a detailed design, and are inappropriate earlier in the lifecycle. Different types of models are needed at Milestone A and Milestone B, before a detailed design is available.

6.2.1 Intellectual Property, Trade Secrets and the Digital Thread/Digital Twin

Understanding of the issues and approaches collaborative engineering, sharing models and data in a *competitive* acquisition environment, and with the need to share models and data that companies consider Trade Secret or Intellectual Property is much less mature. Spreng (2000) surveyed members of the Integrated Dual-use Commercial Companies association regarding barriers to civil-military integration. Protecting IP, including proprietary development and manufacturing processes as well as technologies, was cited as the predominant barrier to collaborative research and engineering development, e.g.,

- Boeing considers the mass properties of the 737 to be IP since the 737 was developed with internal funds during a bid-and-proposal (the USG explicitly states that costs incurred during bid-and-proposal are company cost, and not reimbursable), but Boeing was not selected for award. The 737 is the base platform for many military aircraft. The mass property data has to be exposed in the Digital Twin.

- Hughes Electronics considers their model of friction in electro-mechanical and hydraulic control systems to be IP. The friction model is an essential component of the M1A2 fire control system and other military control systems. The friction model is an essential component of the model of the fire control system that would be in the virtual twin. Hughes Electronics was a vendor supplying the fire control subsystems to General Dynamics, the prime contractor on the M1A2 program, and General Dynamics did not have technical data rights to the friction model and other control system elements that were not developed under contract.

Some of the technical data and models that vendors consider proprietary will have to be exposed, to some degree and to some collaborators, in Digital Engineering Design. In complex engineering programs with multiple tiers of vendors, not all parties need access to all technical data and models, and limited data rights may be sufficient for those that do. When programs involve competitors, e.g., Competitive Prototyping during the Technology Development phase, or second-source development during or after EMD, the situation is even more sensitive. Timing is also a factor. Technical data may have high commercial value when the technologies are novel, but the value of protecting the IP decreases as other technologies emerge.

DoD open architecture initiatives have begun to tackle many of the issues related to Trade Secret and IP data rights and data protection, data exposure requirements, and other engineering collaboration issues (Guertin, 2011). The DoD Open Business Model (McFarland, 2014) provides guidance regarding what technical data needs to be open and how to manage data rights in a competitive acquisition environment. Guertin, Sweeney and Schmidt (2014) summarize lessons learned from Navy open architecture initiatives. Decker (2012) presents a decision process to identify technical data and software deliverables and data rights needed for open architectures.

6.2.2 Pilot Programs

Efforts are underway to update the MIL-STD for the Technical Data Package to reflect the more precise and extensive needs of the Digital Thread (Huang, 2013). This will be an on-going process as the

community gains experience, and the technologies and standards evolve. The many pilot programs underway will provide valuable information to refine the MIL-STD, and inform our project.

- NIST's "Design to Manufacturing and Inspection" demonstration project began in January 2015, and is expected to have preliminary findings and results this year. The project is attempting to demonstrate the feasibility—and benchmark the advantages—of using standardized, three-dimensional (3D) models for electronically exchanging and processing product and manufacturing information all the way from design through inspection of the final part, a tightly integrated, seamless string of activities. The initial goals are to demonstrate a fully automated thread from design though machining. It is part of NIST's ongoing project "Enabling the Digital Thread for Smart Manufacturing."

- The Air Force has three pilot programs under way to develop digital-system modeling for different stages of the acquisition process and demonstrate them on four upcoming procurement programs. The three capabilities are a simulation tool to use pre milestone A, using CREATE-AV compare to wind tunnel tests, third involves digital modeling to determine if non-conforming parts (manufacturing) are acceptable. These efforts show the diversity of tools that are needed for the Digital Twin, and variety of engineering processes that need to be supported. (Tuegel, Kobryn, and Henderson, 2015) (Christian, 2014) (Kobryn, 2014).

- The "Autonomy and Operational Energy" program is TARDEC's pilot program for the Digital Thread/Digital Twin. The program is addressing several concerns that NAVAIR has also expressed:

    - How to address the possibility of "emergent behaviors" and unexpected interactions in distributed, real-time cyber-physical systems over the range of operating conditions

    - How to "re-understand" evidence at each major technical review in a Digital Engineering Design enterprise

    - How to "re-understand" test, evaluation, and acceptance criteria (formally, "Failure Definition and Scoring Criteria") in a Digital Engineering Design enterprise

- Organizations such as NASA/JPL are demonstrating new approaches to fully-driven model-based development using iterative, agile-like approaches (e.g., the Europa project) (Dvorak, 2013), (Cooke, 2015).

6.2.3 Approaches to Share, Protect, and Secure Data in Collaborative Environments

Grimm and Anderl (2013) present an analysis of current technical approaches to protect and secure knowledge in collaborative systems engineering environments. The technical means include knowledge encapsulation exposing only necessary interfaces isolating underlying modes, enterprise rights management for information that is exposed, and security techniques to prevent data exposure outside of the data rights framework.

Tang, Molas-Gallart and Shields (2008) carried out a series of case studies of major United Kingdom collaborative defense projects to identify the problems and approaches to a shared data environment. They developed an "issues and impacts" framework to understand IP issues in collaborative engineering development environments. They distilled two alternative organizational structures and corresponding data rights management strategies: (1) centralized data control backed by bilateral relationships, and

(2) the federated trust environment approach. They conclude that effective management of IP in collaborative programs requires (1) executive leadership to ensure close coordination between contracting, legal, IT and engineering functions, and (2) adaptation to the characteristics of the program. The key challenges they identified were:

- Pre-existing proprietary information that each agent brings to the collaboration that need to be integrated with other technologies in the program
- Convergence of product and process data
- Divergent approaches to IP management and data control among collaborators
- Ad hoc and diverse enterprise solutions across programs

6.2.4 VALUE OF OWNING THE TECHNICAL BASELINE

Not owning the technical baseline for spares has forced increased use of reverse engineering and re-engineering. NAVAIR estimates that they spend approximately $10M per year reverse engineering parts for the F-18, and an average of 9 months for each job. NAVAIR further estimates that 50 percent of the F-18G aircraft are not flyable because NAVAIR does not have the technical data for the parts and/or processes.

Reverse engineering and re-engineering is pervasive across all Service branches. It is a central element of DMSMS programs. The reverse engineering and re-engineering provides a treasure trove of information not just on the reduced operational availability costs of not owning the technical baseline, but also on the time and cost of producing the technical baseline, i.e., reference points for what the technical baseline should cost. The Defense Standardization Program Office (2015) maintains and periodically updates a database of costs and cost models associated with DMSMS resolution. The data are organized by part type (e.g., assembly, component, raw material, software, etc.), commodity type (e.g., electronics, mechanical, electrical, etc.), operating environment (e.g., air, ground, space, etc.), cost type (engineering design, technical manuals, etc.), and cost amount. The database and cost models can help address the question of what should technical data rights cost by what type of data and component. The principle is that buying the data rights in the first place should not cost more than re-creating the technical data.

The difference between the actual and planned lifespan of Defense systems becomes an issue – when short term solutions become long-term platforms. The Technical Baseline becomes more valuable to the USG the longer the system is in service due to DMSMS, and less valuable to the vendors as technologies become obsolete. Of course, there is still value to the contractor from anti-competitive "vendor-lock" if the USG does not own the technical baseline.

Owning the technical baseline enables competition and second sourcing. Hard data on the cost of vendor-lock when the Government does not Own the Technical Baseline, and the cost savings from competition are sparse, but exist. The data are created when a program that was single source is opened to competition, and when, due to a misstep, a vendor obtains exclusive supply rights. Primary research is needed to find these cases and compile the data.

> *The case of the M4 carbine. In 1967 Colt and the USG entered into a patent agreement on the M16 rifle. In 1985, the agreement was extended to the M4 derivative, but the license agreement gave the Government limited rights to the technical data and placed restrictions on its use. The license agreement included a provision that limited the Army's right to transfer or release the*

technical data package. The USG was later found to have inappropriately released the information to competitors. The matter was settled with the USG granting a 10-year exclusive license, at which point contractor raised the unit price from $512 to $912, and ultimately to $1,012. After the 10 years expired, Colt reduced the price to $815. In 2012 a new competition was held in which Colt and the Army agreed to a 5% royalty for every carbine obtained from a second source, and added the royalty to the second source's competitive bid. The contract for the M4 was awarded to Remington for $673 (unit production cost of $641 plus the 5% royalty for use of the data rights). In this case, vendor-lock produced 80% growth in unit cost. The threat of competition produced approximately 20% reduction in unit cost. Actual competition further reduced the unit cost by approximately 20% (before the 5% royalty for use of the data rights). (Watters, 2011)

Owning the technical baseline and open architectures enable competition, and competition impacts development program performance. The JDAMS program estimated that dual sourcing saved 33 percent in development time, 42 percent in development cost, and 50 percent in unit production cost (Wydler, Chang and Schultz, 2013).

The Digital Thread / Digital Twin vice a "document centric" TDP will improve competitiveness in contracting for upgrades and second-sourcing. The recent competition for the Armored Multi Purpose Vehicle (AMPV) is a case in point. The AMPV was planned to be built using the Bradley chassis. The USG delivered the massive and outdated paper TDP to all bidders. Only one company submitted a bid, the company that developed the Bradley. The other potential bidder dropped out, citing that the year allowed was insufficient time to digest the data.

6.2.5 METHODS TO VALUE IP EXPOSURE

Current methods to value data rights to IP and trade Secrets may need to be adapted to apply in the context of collaborative engineering environments, and competitive acquisition under conditions of monopsony when the intent of the buyer is to create competition. Head and Nelson (2012) summarize relevant best practices that balance cost-and-value to the vendor and customer. Reilly, Garland, and Zanni (2009) describe the alternative methods and considerations in valuation of intangible assets over the lifecycle of the assets.

6.2.6 REVERSE-ENGINEERING AND RE-ENGINEERING

Reverse-engineering and re-engineering are pervasive practices in situations of Diminishing Manufacturing Sources and Materiel Shortages (DMSMS). DoD has Manufacturing Technology (ManTech) initiatives in all Service Branches addressing DMSMS. Reverse-engineering refers to producing the TDP for an existing component. Re-engineering refers to producing a TDP for a component with the same "form, fit, function, operation, and maintenance" – which the USG has unlimited rights to, provided the data are required deliverables.

Unavailability of "spare parts" (subsystem and components) to maintain existing military systems is a significant problem. NAVAIR estimates that the cost of reverse engineering parts for the F-18 and the V-22 cost $10M per year for each program, and nine months, on average, to re-engineer the parts. Even more costly from an Operation and Sustainment (O&S) perspective is the estimate that 50-percent of the aircraft are not flyable because NAVAIR does not have the spare parts due to lack of technical data on parts and/or processes.

Reverse-engineering and re-engineering is an increasing O&S cost footprint. While this is an unhappy outcome from a Better Buying Power perspective, these data provide evidence of the cost and value of owning the Technical Baseline: the cost of re-creating the technical baseline when it is needed is a floor on the cost, and the cost savings is a ceiling on the value. Agencies engaged in re-engineering and reverse engineering (e.g., Warner Robbins AFB) have historical data on the time and cost to re-engineering components and systems (including complex electronics and software-intensive systems) vice operational, physical and logical characteristics. Compiling the data had a cost.

"Vendor depth" is an issue when a higher-tier supplier employs subsystems/components from lower-tier suppliers, especially using Commercial Off-the-Shelf (COTS) components. The "fine print" of many COTS products prohibits dis-assembling and re-engineering or reverse engineering. For software and behaviors of cyber-physical subsystems, the "form, fit and function" is indistinguishable from the design. When the USG buys a system, subsystem, or component is has unlimited rights to know the "form, fit, function, operation and maintenance" (provided the rights are claimed in the contract). These rights are essential for Operation and Sustainment, to reverse-engineer or re-design components that the original supplier no longer supports.

6.2.7 Value of the Digital Thread and Digital Twin

Data on the value of the Digital Thread/Digital Twin – development time and cost reduction, quality improvement, maintenance turn-around time, ability to secure spare parts, etc. – is beginning to accrue as Service agencies begin to implement Digital Engineering Design practices. The ARL USG-Industry Working Group (Huang, 2013) estimated that:

- When manufacturers use 3D models, they build only half the number of prototypes
- 3D tools reduce the development cycle by 30-50 percent
- Standard parts libraries provide significant reduction in component assembly time (design time)
- 3D models reduce non-conformance issues by 30-40 percent
- 40 percent of non-conformances are due to 2D drawing inaccuracies and ambiguities
- 85 percent of companies still use 2D drawings in their operations or with their suppliers

6.3 Approach and Rationale

The elements of our technical approach are:

1. To continue to engage and coordinate with stakeholders across DoD, Defense Industrial Base, and Dual-Use, including participation in the annual NIST Model-Based Enterprise Summit
    a. To remain cognizant of related DoD and Industry priorities and initiatives in order to maintain consistent frameworks and lexicon, to share and leverage findings and results, to ensure practicality and relevance
    b. To identify transition and execution issues, gaps and approaches from different perspectives across DoD, and the Industrial Base
    c. To facilitate diffusion and approval
2. To complete the literature review begun in the ignition phase addressing
    a. Applicable DoD and Defense Industry policies, claims and practices
    b. Emerging DoD and Industry Digital Engineering Design and collaborative Systems Engineering

    c. Defense Industry contractual and business impact concerns
    d. Technologies, methods and policies regarding collaborative engineering and engineering environments in Digital Engineering Design
3. To develop the reference model, objective "A" as the concepts, expectations, requirements, and specifications for the Digital Thread and Digital Twin evolve
    a. Leveraging findings from (1) and (2) above, and findings from related SERC research tasks RT118/141 and RT148
    b. Develop in incremental stages
        i. Year 1: Focus on near-term opportunities and long-term transition framework for the Digital Thread and Digital Twin, and coordinate the "Phase 1" reference model
        ii. Year 2: Focus on refinement, incorporating results of pilot programs and reflecting developments in related initiatives, and coordinate the "Phase 2" reference model
        iii. Year 3: Focus on refinement, incorporating results of pilot programs and reflecting developments in related initiatives, and coordinate the "Phase 3" reference model
    c. Demonstrate by example application to a Digital Engineering Design pilot program, in collaboration with the executing agency acquisition agency
4. To develop the supplemental information, objective "B"
    a. Identify current and emerging standards and approaches
    b. Identify near-, mid- and long term issues and gaps from (1), (2) and (3)
    c. Identify transition risks
    d. Continue to collect evidence of the costs and benefits of the digital TDP and "Owning the Technical Baseline"
    e. Map the information to the reference model
5. To develop MPT for planning an incremental transition strategy, objective "C"
    a. Locating groupings of connected near-term (mid-term) issues and gaps that, if solved, would provide significant increases in Digital Engineering Design capability and diffusion
    b. Identifying long-term issues and gaps that limit or prevent significant increases in Digital Engineering Design capability and diffusion
    c. Analyzing the reference model to locate nodes and transitions that have widespread impact on collaborative Digital Engineering Design capability
6. Technical reviews
    a. Quarterly or semi-annual reviews with the community of interest
    b. Annual SERC and SERC-sponsor technical reviews

We have practical and first-hand experience in all of these issues. Some of these issues involve IP and Defense contracting, legal, and negotiation concerns. We will work with the *Levin Law Center* at Wayne State University, led by former Senator Levin, former Chairman of the Senate Armed Services Committee, now retired on these issues that were of great significance while he was in office. We have historical and on-going collaborations with the members.

Research during the initial investigation, and in related RTs, provides evidence that this is a practical and feasible approach.

We have engaged effectively with the Defense Industry, Government agencies regarding model-centric engineering opportunities, challenges and practices. The defense industries have been willing to talk candidly with us because we are not trying to sell them anything, and because it is seen as an avenue to have an impact on the future shape of model-centric acquisition.

We have identified several programs engaged in piloting Digital Engineering Design (in one form and degree or another). These include the NIST Design to Manufacturing and Inspection pilot project as part of the NIST Digital Thread for Smart Manufacturing initiative. The USAF has initiatives underway to pilot Digital Twin approaches at three different stages of the acquisition lifecycle.

We have identified initiatives to develop standards and approaches to collaborative Digital Engineering Design, under RT118/141. We have found significant literature on technical data rights issues in model-centric collaborative engineering. We have been able to identify sources of data and evidence on the cost of developing a TDP, the time and defect-rate savings of a digital TDP, and the cost-of vendor-lock and savings from competition, and valuation of Trade Secrets and IP.

We have experience mapping engineering workflows and model requirements, and using network analysis to identify key gaps and activities.

6.4 WHO CARES? CUSTOMER SEGMENTS AND VALUE PROPOSITIONS

This project was proposed in response to a challenge from Dr. David Gorsich, Chief Scientist at TARDEC:

> *My greatest need for SE research is to understand how to value IP and Tech Data, how to convince the PMs to pay for a complete TDP, how to get what we need in the TDP as we move to more integrated model centric acquisition, how to compensate industry for IP and "trade secret" engineering processes that may be exposed when sharing models and data.*
>
> *We need to break vendor lock, and used competition to drive down cost and drive up quality. We can't do that if we don't own the models and data.*
>
> *We also want to reduce acquisition time and defects by collaborative development with industry, with open-source development and model-vs-paper exchange. But industry has to be on board. And they won't be if they think they are giving away competitive advantage.*
>
> *Where can we get data? Without it, we don't know, and can't make a case.*

The project formulation was influenced by insights from David Cohen, Director of Systems Engineering at NAVAIR: (1) the need to execute hundreds of parallel and interacting activities during engineering development, (2) the need for comprehensive use of modeling and simulation (M&S) throughout the system lifecycle, to make extensive use of virtual testing do identify downstream problems early in the program, and (3) the need for new practices and tools for USG and contractors to interact in model-driven collaboration. He asked an insightful question that we will try to explore: "*Does Digital*

Engineering Design and collaborative engineering in a competitive acquisition environment offer new options and possibilities for system acquisition? If so, what are they and how do we get them to accelerate system development and enhance sustainment?"

The project is coordinating its products and objectives with the DASD Digital Engineering Design initiative led by Phil Zimmerman. One of the two major branches of DASD(SE) is Engineering Enterprise. One of the groups within the Engineering Enterprise branch is Engineering Tools and Environments. Digital Engineering Design is one of the three pillars of Engineering Tools and Environments, the other two being Engineered Resilient Systems, and Modular Open Systems Architecture. The main thrusts of the Digital Engineering Design activity are: Digital System Model/Digital Thread, Education, Policy and Guidance, Data Rights. DASD(SE) collaborators include NASA, CAPE, Army, Navy and Air Force agencies, and the NDIA M&S subcommittee. (Zimmerman, 2015).

Many DoD agencies are working to transform their "as is" enterprise to some form of Digital Engineering Design, are working to develop guidelines for that end state, and a transformation plan. TARDEC and NAVAIR both have such initiatives. This project will help complete the vision of a collaborative engineering environment for Digital Engineering Design by addressing the data rights identification and management issues and approaches.

The products of the proposed research will assist DoD agencies in developing plans to transition to and operate in collaborative digital environments, in establishing frameworks and data right management that will enable USG, industry competitors and suppliers to share models and data, and to convince PMs of the value relative to costs.

6.5 Transition Plan, Outcomes and Impacts

Our transition plan is to engage with collaborators with at DASD, TARDEC and NAVAIR throughout the project. We also plan to engage with Defense Industry, other DoD agencies, and other USG agencies (NIST, NASA/JPL) via the annual NIST MBE Summits to socialize the our finding and results.

6.6 Risks and Opportunities

Risks:

- Relevant cost data may be insufficient for statistical confidence, and only give examples of costs and savings from the digital TDP, competition, collaboration, etc. Mitigation: NAVAIR have been collecting cost data. The Defense Standardization Program Office has been assembling a database of re-engineering and reverse engineering cost, frequency and impact.
- Policies and practices for collaboration and digital engineering are evolving, as are technical standards and technologies, which will impact costs and savings. Mitigation: We will coordinate with these initiatives.

Opportunities:

- The products address clearly articulated needs from the Defense acquisition community, both DoD and the Defense Industry

- The Defense Industry majors – Boeing, Lockheed Martin, Northrup Grumman, General Dynamics, etc. – are moving rapidly to implement their versions of Digital Engineering Design practices. The products of this research will help DoD agencies avoid being boxed in to accepting contractor standards and methods because of their maturity
- The project is timely and its schedule is "in sync" with related initiatives, forums, demos and pilot studies. There is an opportunity to influence evolving policies and practices both at DASD(SE) and acquisition commands. There is an opportunity to make use of data from demos, and pilot studies.
- This project will leverage findings and results from RT118/141 (with NAVAIR) and RT148 (with TARDEC)

6.7 Cost and Schedule

- Year 1: $200K. Initial reference model for the Digital Thread, and supplementary data
- Year 2: $180K. Refined reference model, extended to include the Digital Twin, with enhanced supplementary data – coordinated with external pilot programs and initiatives
- Year 3: $160K. Refined reference model, extended to the transition model and planning tool, with enhanced supplementary data

6.8 Progress Measurement and Milestone Tests

Progress will be measured by annual technical reviews assessments made by key stakeholders at DASD(SE), TARDEC, and NAVAIR. The technical reviews will provide a technical description of the reference mode, supporting data, and the MPT for transition planning. The technical reviews will include findings and results from the example application of the research products in an example case study.

References

Blackburn, M., R. Cloutier, G. Witus, E. Hole, M. Bone, Transforming System Engineering through Model-Centric Engineering, SERC-2014-TR-044-2, January, 2015.

Christian, T. Determining the Contents of the Digital System Model. AFRL. NIST MBE Summit. 2014.

Cooke, B. Model-based Systems Engineering Mission Formulation on the Europa Clipper Pre-Project, NASA - JPL / California Institute of Technology, NASA/JPL Symposium on Model Based Systems Engineering, January, 2015.

Decker, W. Identifying Needed Technical Data and Software. Defense Acquisition University. 2012.

Defense Standardization Program Office. Diminishing Manufacturing Sources and Material Shortages Cost Metrics. 2015.

Dvorak, D. Model-Centric Engineering, Part 1. NASA. 2013.

Grimm, M., and Anderl, R. Intellectual Property Protection and Secure Knowledge Management in Collaborative Systems Engineering. Procedia Computer Science 16. 2013.

Guertin, N. Open Systems Architecture and Data Rights Overview. US DoD. 2011.

Guertin, N., Sweeney, R., and Schmidt, D.. How the Navy Is Using Open Systems Architecture to Revolutionize Capability Acquisition. NDIA conference. 2014.

Head, S., Nelson, J. Data Rights Valuation in Software Acquisitions. Center for Naval Analysis. 2012.

Huang, P. Technical Data Package Specification for 3D MBD and MIL-STD-31000 Overview. NIST MBE Summit. 2013.

Kobryn, P. MBE & the Digital Thread. AFRL. NIST MBE Summit. 2014.

McFarland, K. Open Business Model for Unmanned Aircraft Ground Control Stations. DoD AT&L. 2014.

Spreng, R., The Intellectual Property Barriers To Civil-Military Integration: A Survey and Recommendations. IDCC Integrated Dual-use Commercial Companies publication. 2000.

Reilly, R., Garland, P., and Zanni, K. Valuation of Intangible Assets for Fair Value Accounting Purposes. NACVA Illinois State Chapter Meeting. 2009.

Tang, P., Molas-Gallart, J., and Shields, R. Collaborate But Protect: The Challenges of Protecting Your Data. INGENIO Working Paper Series. 2008.

Tuegel, E., Kobryn, P., and Henderson, D. Developing the Airframe Digital Twin. AFRL 2015.

Watters, D. Colt M4 Data Rights & The Individual Carbine Competition. Defense Industry Daily. 2011.

Wydler, G., Chang, S., and Schultz, E. Continuous Competition as an Approach to Maximize Performance. Defense Acquisition University. 2013.

Zimmerman, P. MBSE in the Department of Defense. Goddard Space Flight Center Seminar. May, 2015b.